The TOMCAT/SLIMCAT Off-Line 3D CTM User's Manual

Martyn Chipperfield

Institute for Atmospheric Science
School of Earth and Environment
University of Leeds

Volume I

A Description of the Model and a Guide to Using it

Draft - Still being written!!

August 2006

Version 0.8

Contents

1 Introduction **4**

2 Background **4**
2.1 History . . . 4
2.2 Web Pages . . . 5
2.3 Nupdate and Libraries . . . 5
2.4 Fortran . . . 6
2.5 Parallelisation . . . 6
2.5.1 OpenMP Parallelisation . . . 7
2.5.2 MPI Parallelisation . . . 7
2.6 Platforms . . . 7

3 Basic Dynamics **8**

4 Model Grid **9**
4.1 Horizontal grid . . . 9
4.2 Vertical Grid . . . 9

5 Transport and Dynamics **10**
5.1 Mass Fluxes . . . 10
5.1.1 Consistency of Vertical/Horizontal Mass Fluxes . . . 10
5.1.2 Consistency of Advected Mass and Analyses . . . 10
5.2 Advection: The Prather Scheme . . . 10
5.3 Advection: The Semi-Lagrangian Scheme . . . 10
5.3.1 Extended Grids . . . 11
5.3.2 Interpolation . . . 11
5.3.3 Coordinate Transformation . . . 11
5.4 Other Advections Schemes . . . 11
5.5 Trajectory Calculations . . . 11

6 Physics **12**
6.1 Convection . . . 12
6.1.1 Convective Updraft . . . 12
6.1.2 Subsidence . . . 16
6.2 PBL . . . 17
6.3 Radiation . . . 17

7 Chemistry Schemes **18**
7.1 Stratospheric . . . 18
7.2 ASAD-based 'Tropospheric' . . . 18
7.3 Cariolle Ozone . . . 18
7.4 User Supplied . . . 18
7.5 GLOMAP . . . 19
7.6 DLAPSE . . . 19

8 Forcing Winds and Temperatures **20**
8.1 ECMWF Analyses . . . 20
8.2 UK Met. Office UARS Analyses . . . 20

9 Running the TOMCAT/SLIMCAT Model **21**
9.1 Parameters . . . 21
9.2 Model Switches and Variables . . . 21
9.3 Fortran Channels . . . 23
9.4 Subroutines . . . 23
9.5 Advection . . . 24
9.5.1 Prather Scheme . . . 24
9.5.2 SLT Scheme . . . 24
9.6 Output Files . . . 25
9.6.1 PDG . . . 25
9.6.2 DIAG . . . 25
9.6.3 STAT . . . 25
9.6.4 ZON . . . 25
9.6.5 OXO . . . 25
9.6.6 REST . . . 25

10 Acknowledgements **25**

11 Appendix 1. Notation **28**

12 Appendix 2. Universal Constants **30**

13 Appendix 3. Details of the Prather Advection Scheme **31**
13.1 Definitions . . . 31
13.2 Principle of scheme . . . 33
13.3 Generalisation of the Scheme . . . 35
13.4 Use in Spherical Geometry . . . 36
13.5 Correction of Negative Values . . . 41

14 Appendix 4. Chemically Updating Tracer Moments **42**

15 Appendix 5. MPI version of TOMCAT/SLIMCAT **45**
15.1 Domain Decomposition . . . 45
15.2 Defining the Grid . . . 46
15.3 Compiling and Running . . . 47
15.4 MPI Performance . . . 47

16 Appendix 6. Writing Fortran for TOMCAT/SLIMCAT **49**
16.1 General Rules in Fortran . . . 49
16.1.1 When Writing Code . . . 49
16.1.2 When Testing Code . . . 49
16.2 Styles/Conventions in TOMCAT/SLIMCAT: . . . 50
16.3 Example Subroutine . . . 50
16.4 Example Interface . . . 51

17 Appendix 7. Flowtrace and other performance tests **53**
17.1 TOMCAT - Cray YMP8 53
17.2 SLIMCAT - Cray YMP8 54
17.3 SLIMCAT - Cray YMP8 55
17.4 TOMCAT - Green O3000 56

1 Introduction

This manual describes the TOMCAT/SLIMCAT off-line three-dimensional (3D) chemical transport model (CTM). The manual is aimed at providing both some background to the formulation/approach of the model as well as some instruction on how to run it (or at least understand the standard job decks).

Section 2 gives some background to the model. Section 3 provides some basic definitions of atmospheric dynamics. Section 4 describes the model grid. Section 5 describes the model transport and advection. Sections 6 and 7 describe the model physics and chemistry routines, respectively. Section 8 provides information on the sources of winds available to force the model. Section 9 gives information on actually running the model. Users simply interested in running the model could just focus on Sections 2, 7, 8 and 9.

2 Background

2.1 History

The TOMCAT and SLIMCAT off-line CTMs have been widely used for atmospheric chemistry studies over the past decade or more. The TOMCAT[1] model was originally written in 1991/92 at the Centre National de Recherches Météorologiques (CNRM), Toulouse and was first described and used by *Chipperfield et al.* [1993] for studies of the polar stratosphere. The TOMCAT model used hybrid σ-p levels. Although it performed reasonably well it was not ideal for stratospheric studies and could not make best use of the stratospheric forcing analyses then available. Therefore, I developed the related SLIMCAT 3D CTM - first described in *Chipperfield et al.* [1996]. This differed from TOMCAT in using an isentropic vertical coordinate (and hence having a domain effectively limited to the stratosphere) and using diagnosed heating rates to derive the vertical transport. These developments allowed the model to use the UKMO analyses for multiannual stratospheric simulations (e.g. Chipperfield, 1999). Meanwhile, the TOMCAT CTM was further developed for tropospheric chemistry studies, by the inclusion, for example, of convection [*Stockwell and Chipperfield*, 1999], wet and dry deposition [*Giannakopoulos et al.*, 1999], lightning [*Stockwell et al.*, 1999] and detailed gas-phase chemistry [*Law et al.*, 1998]. Effectively SLIMCAT became a stratospheric CTM and TOMCAT, usually forced by ECMWF analyses which extended to only 10 hPa, a tropospheric CTM.

Recent technical and scientific developments have meant that it would be desirable to combine the two-related CTMs into a single model with a flexible vertical coordinate and different methods or treating key processes (e.g. vertical transport). First, in 1999 the ECMWF extended the upper boundary of their analyses to 0.1 hPa and recently completed the ERA-40 reanalyses from 1957-2002 over this same domain. Second, many key scientific issues concern processes in the upper troposphere / lower stratosphere and so model boundaries in this region are not desirable.

This manual describes the new version of TOMCATi/SLIMCAT which combine both original models (and other routines) into a single library. The resulting model extends from the surface upwards with different options for the vertical coordinate. The new library can still be used for all past applications of the earlier models with some significant new possibilities.

[1] TOMCAT was originally an acronym for the Toulouse Off-line Model of Chemistry and Transport, though now it is just a name

2.2 Web Pages

The home web page for the CTM is at:

`www.env.leeds.ac.uk/tomcat`
`www.env.leeds.ac.uk/slimcat`

Both addresses link to the same page! On these pages are general information about the models, an up-to-date-list of references which have used the models (with reprints of some papers), links to results for recent campaigns etc.

There are also pages with restricted access for Users and Leeds-based people. Please get password from me.

2.3 Nupdate and Libraries

The TOMCAT/SLIMCAT code is contained in a library which is managed using 'nupdate'. Nupdate is software originally developed for Cray supercomputers and provides a nice environment for maintaining a centralised, standard version of a model while still letting individual users make temporary changes in their own experiments. Personally, I think nupdate is still the best and easiest way of keeping control over large programs.

With nupdate the basic model library exists as a series of 'decks' and within each deck every line of model code has a unique line number. When you run a job you combine selected decks from the standard library, with temporary changes to individual lines and nupdate produces the final Fortran program which can then be compiled.

The current TOMCAT/SLIMCAT library is contained in the file 'UNICAT0.8'. This file cannot be edited and so there is also a listing file ('`unicat0.8_list`') which can be viewed and used to identify line numbers (see `martyn/BIBLI/unicat0.8_list`). An example of the listing file is:

```
*COMDECK,PARADI                                                    PARADI.1
C                                                                  PARADI.2
      INTEGER ITRIG0,ITRIG1,IIFAX0,IIFAX1,LAT                      PARADI.3
      INTEGER LATS2,LEV,LIGMAX0,LKGMAX0,LKG2,LOLA,LON,LPRO         PARADI.4
      INTEGER MI0,MI1,MLON,MLAT,MP0,MP1                            PARADI.5
      INTEGER NIV,NMP0,NMP1,NMI0,NMI1,NPAR,NPVAR,NTRA,NVRJPJ,NVTOT PARADI.6
C                                                                  PARADI.7
C  parametres principaux du modele                                 PARADI.8
C                                                                  PARADI.9
      PARAMETER(LON= 64,LAT= 32,NIV= 2)                            PARADI.10
      PARAMETER(NTRA=25,NVTOT=35)                                  PARADI.11
      PARAMETER(NVRJPJ=0)                                          PARADI.12
      PARAMETER(LOLA=LON*LAT,LPRO=LON*LAT*NIV,LATS2=LAT/2)         PARADI.13
```

The user can make temporary changes to the standard library by using the following nupdate commands:

Command	Meaning
`*I,deck.xx`	Insert after line `deck.xx`
`*B,deck.xx`	Insert before line `deck.xx`
`*D,deck.xx,yy`	Delete the range of lines `deck.xx` to `deck.yy` (and insert any lines which follow)

The user can also control which decks are included in the model Fortran file. This avoids the need to include (and compile) every line in the library in every job and is also a way in which alternative parameterisations can be selected. The nupdate decks are selected by using '`*C`'. Therefore, part of a typical TOMCAT/SLIMCAT job may start:

```
cat <<'eof' > tom1.up
*C,SLIMCAT,INIEXP,INICSTE,INICYCL,INITER,FINITER,FINCYCL,FINEXP
*C,ALRET,CALFLU,CORPOLE,CHTRUK,CHTRON,CALNUM,RADIAT

<<+ select other decks needed>>

*IDENT,EXP
*D,PARADI.10,11
C     Model grid
      PARAMETER(LON= 64,LAT=32,NIV=31)
C     Number of tracres
      PARAMETER(NTRA=30,NVTOT=48)

<<+ other temporary updates>>

eof
```

The official version of Cray nupdate does alot more than this, but it is also very difficult to install on a system. Therefore, a stripped-down version written in plain f77 specifically for the CTM is available. This has a limited set of the nupdate commands and is easy to use on any platform (it just needs a Fortran compiler) and makes the model easily portable.

Overall there are 3 nupdate libraries relevant for TOMCAT runs. In addition to the TOMCAT/SLIMCAT library itself there is a library containing the ASAD chemical solver (`UNIASADx.x`, listing in `uniasadx.x_list`) and a library for the CCM boundary layer scheme (`UNIPBLx.x`, listing in `unipblx.x_list`).

2.4 Fortran

The original TOMCAT and SLIMCAT models were written in quite standard Fortran77. On the basis that there is no point rewriting something which works fine, these routines are still Fortran77. Some newer additions to the model library use Fortran90 (e.g. the chemical data assimilation code; the DLAPSE microphysical model). Depending on the components selected for a particular run then a f77 compiler may be ok. An f90 compiler will be ok in all cases.

Some notes on writing Fortran code for TOMCAT/SLIMCAT are given in Appendix 6.

2.5 Parallelisation

TOMCAT/SLIMCAT was originally written to run efficiently on vector machines (e.g. old Cray machines). This is achieved by making inner DO loops vectorisable - and the code should still conform to that. However, the high performance computers (HPCs) available for running the model now are parallel machines.

There are different methods for parallelising code. These methods can be divided into 'distributed memory' or 'shared memory' categories. In distributed memory machines a copy of the code is run on each processor and when necessary messages are passed between the processors to communicate data. This approach requires that the code be specifically written for the parallel machine using, for example, Message Passing Interface (MPI). In shared memory machines the code is run on all the processors which work on a subset of the common memory. This approach is usually much easier to implement in existing codes, e.g. using OpenMP.

2.5.1 OpenMP Parallelisation

The standard version of TOMCAT/SLIMCAT has been parallelised using OpenMP. In OpenMP work is generally shared at the 'DO LOOP' level. As such existing codes can be parallelised in stages, e.g. so that only the most expensive loops are shared over all the processors and the rest of the code is just run on a single processor (the Master thread). The disadvantage of OpenMP is that the efficiency of the parallelisation of individual loops may not scale that well with increasing numbers of processors. To get good parallel speed-up the parallelisation should be done at the highest level possible, e.g. the chemistry is parallelised over an outer 'latitude' loop which contains all the routines to solve the chemistry on a longitude×level slice. With this the parallelisation speedup for the chemistry (an expensive part of the model) is effectively 100% efficient. The speedup is similar for other expensive parts of the model, such as the radiation schemes and boundary layer scheme, which have an outer latitude loop which can be easily parallelised. In effect it is only the dynamical subroutines where the speed-up is not as efficient. A consequence of this method of parallelisation is that the number of CPUs used to the run the model should be a factor of the number of model latitudes (`LAT`) for optimum efficiency.

For a scalar (non-OpenMP) job the OpenMP directives inside the code are ignored.

2.5.2 MPI Parallelisation

Appendix 5 describes an MPI version of TOMCAT/SLIMCAT which is available for use on HPC machines which do not support OpenMP or for which MPI jobs can use more processors (e.g. on HPCx OpenMP jobs are limited to 16 CPUs). The MPP library is equivalent to the standard library and allows the major options to be run (e.g. full stratospheric or tropospheric chemistry runs).

2.6 Platforms

In the past TOMCAT/SLIMCAT has been successfully run on a wide range of machines, e.g. Cray (2, YMP, J90), Hitachi, Fujitsu, Sun workstations, SGI workstations and linux PC (including my laptop). Currently the default machines for the model runs are the SGI O3000 machines at CSAR (Wren/Green/Fermat).

At HPCx the libraries and standard jobdecks for TOMCAT/SLIMCAT are located in `/hpcx/home/n02/n02/lrkd/unicat`. There are readme files which list the files there.

3 Basic Dynamics

This section lists some basic equations of atmospheric motion which are used in the detailed description of the transport part of TOMCAT/SLIMCAT (Section 5). The symbols used are summarised in Appendix 1.

The continuity equation is

$$\frac{\partial}{\partial \eta}(\frac{\partial p}{\partial t}) + \nabla.(\vec{\mathbf{v_h}}\frac{\partial p}{\partial \eta}) + \frac{\partial}{\partial \eta}(\eta\frac{\partial p}{\partial \eta}) = 0 \tag{1}$$

Integrating this equation from the top of the atmosphere to the ground gives the rate of change of surface pressure:

$$\frac{\partial p_s}{\partial t} = -\int_o^1 \nabla.(\vec{\mathbf{v_h}}\frac{\partial p}{\partial \eta})d\eta \tag{2}$$

Integrating the continuity equation (1) from the top of the atmosphere to level η gives the equation for the vertical velocity. In pressure coordinates this is:

$$\omega = \frac{\partial p}{\partial t} = -\int_o^\eta \nabla.(\vec{\mathbf{v_h}}\frac{\partial p}{\partial \eta})d\eta + \vec{\mathbf{v_h}}.\nabla p \tag{3}$$

In hybrid vertical coordinates the vertical velocity is:

$$\eta\frac{\partial p}{\partial \eta} = -\frac{\partial p}{\partial t} - \int_o^\eta \nabla.(\vec{\mathbf{v_h}}\frac{\partial p}{\partial \eta})d\eta \tag{4}$$

The divergence is defined as:

$$D = \frac{1}{a}\left(\frac{1}{1-\mu^2}\frac{\partial U}{\partial \lambda} + \frac{\partial V}{\partial \mu}\right) \tag{5}$$

The relative vorticity is defined as:

$$\zeta = \frac{1}{a}\left(\frac{1}{1-\mu^2}\frac{\partial V}{\partial \lambda} + \frac{\partial U}{\partial \mu}\right) \tag{6}$$

The streamfunction (ψ) and the velocity potential (χ) are given by:

$$U = \frac{1}{a}\left\{-(1-\mu^2)\frac{\partial \psi}{\partial \mu} + \frac{\partial \chi}{\partial \lambda}\right\} \tag{7}$$

$$V = \frac{1}{a}\left\{\frac{\partial \psi}{\partial \lambda} + (1-\mu^2)\frac{\partial \chi}{\partial \mu}\right\} \tag{8}$$

$$\zeta = \nabla^2\psi \tag{9}$$

$$D = \nabla^2\chi \tag{10}$$

4 Model Grid

4.1 Horizontal grid

The model grid is variable and determined usually by computational considerations. The longitudinal spacing is regular although the latitudinal spacing can be irregular.

4.2 Vertical Grid

Many GCMs are formulated using a hybrid $\sigma - p$ vertical coordinate. In this scheme the model levels vary from pure terrain-following σ levels near the surface to pure pressure levels at higher altitudes (typically above ~100 hPa). The pressure (p) of a model half-level (interface) is then given by:

$$p_{k+\frac{1}{2}} = Ap_s + Bp_o \tag{11}$$

Consequently many datasets of meteorological analyses are also produced on these hybrid $\sigma - p$ levels. Appendix 1 defines the symbols used in this report.

In TOMCAT/SLIMCAT the vertical coordinate of the model levels can be defined in two ways: (i) hybrid $\sigma - p$ levels or (ii) hybrid $\sigma - \theta$ levels. For hybrid $\sigma - p$ levels the definition of the levels follows equation (11). For the hybrid $\sigma - \theta$ levels the definition of the model levels changes with altitude. Above a reference potential temperature (θ_0) the model uses pure isentropic levels and the potential temperature of an interface is defined as:

$$\theta_{k+\frac{1}{2}} = C\theta_0$$

Between θ_0 (e.g. typically 350 K) and the surface the model uses hybrid levels. The pressure of the model half levels in this region is then given by:

$$p_{k+\frac{1}{2}} = Cp_{\theta 0} + Dp_s$$

where $p_{\theta 0}$ is the pressure at the lowest purely isentropic half level and $D = 1 - C$. This method of defining a vertical coordinate is possible in an off-line CTM as p at the surface (p_s) and at θ_0 are known at future times (using p_s and T from the analyses).

[[Bottom boundary...]]

5 Transport and Dynamics

A key part of any off-line CTM is the routines which convert the wind/temperature analyses used to force the model to the CTM grid and the tracer advection schemes. The conversion of the wind/temperature fields should be done in such a way that the analyses are not degraded or modified unduly. The tracer advection method should ideally have the desirable properties of conservation, low numerical diffusion, etc.

This section first describes the way in which TOMCAT/SLIMCAT maps the analysed wind fields onto the model grid. The following subsections described the main tracer advection routines available in the model.

5.1 Mass Fluxes

5.1.1 Consistency of Vertical/Horizontal Mass Fluxes

5.1.2 Consistency of Advected Mass and Analyses

In SLIMCAT the total mass is an advected quantity. After 1 advection step this mass should ideally be equal to that diagnosed from the analyses used to force the model. However, as the vertical mass fluxes are diagnosed from a radiation scheme, and due to spatial/temporal interpolation errors, this will not be the case.

It is possible to correct the mass fluxes within SLIMCAT so that..

Mass above a (theta) level

$$psup(l) = \int_{l}^{top} plt(l)/ggdphidlambda \tag{12}$$

The mean mass flux over the cycle:

$$wgrs(l) = \int_{l}^{top} 0.5(wgri_1 + wgri_2)dphidlambda \tag{13}$$

Mass change above model interface over cycle:

$$dm = psup_2 - psup_1 \tag{14}$$

Correction to integrated mass flux:

$$corr = dm/dt \tag{15}$$

which is spread evenly over surface:

$$dwgri = corr/4\pi rr \tag{16}$$

5.2 Advection: The Prather Scheme

The default advection scheme used in TOMCAT/SLIMCAT is a gridpoint scheme which is based on the conservation of second order moments of the tracer distribution within the gridboxes. This scheme possesses the properties of being stable, accurate, conservative and is one of the best performing of the transport schemes available. The second order moments scheme was developed by *Prather* [1986]. The model can also be used with the conservation of first order moments (equivalent to the slopes scheme of *Russell and Lerner* [1981]) or conservation of only zero order moments.

The Prather advection scheme is described in detail in Appendix 3.

5.3 Advection: The Semi-Lagrangian Scheme

The SLT scheme implemented in TOMCAT/SLIMCAT is based on the method of *Williamson and Rasch* [1989] and closely follows the details given in their paper.

5.3.1 Extended Grids

The SLT scheme requires the wind velocities and tracer mixing ratios to be known on a grid which extends two gridpoints beyond the standard model grid in the X,Y, and Z directions. This is required by the cubic interpolation scheme. Cyclic continuity is used in longitude. In latitude, the grid is extended to include a pole point (row) and one row across the pole. In the vertical linear interpolation is used in the upper and lower two levels to avoid the need to create points outside the model domain.

5.3.2 Interpolation

The interpolation method has an important impact on the performance of a semi-Lagrangian advection scheme. In the scheme described here the wind velocities and tracer mixing ratios are interpolated in each of the X, Y and Z directions using a cubic Lagrange scheme.

5.3.3 Coordinate Transformation

For trajectory positions polewards of `DLATLIM` (see below) the SLT scheme performs a coordinate transformation to a local geodisic coordinate. The local geodisic coordinate is essentially a rotated spherical coordinate system whose equator goes through the original poles. This is based on the transformation described by Williamson and Rasch [1989] except only one transformation is performed, rather than a different transformation for each trajectory point.

5.4 Other Advections Schemes

[[WAF]]

5.5 Trajectory Calculations

TOMCAT/SLIMCAT can calculate Lagrangian air parcel trajectories. The model can either be used for just this purpose, or Langrangian trajectories can be included in a normal 'Eulerian' TOMCAT/SLIMCAT run for comparison. The same options for trajectories exist as for the main CTM, e.g. vertical coordinate, method for vertical motion.

The trajectory code uses either a simple Euler forward time scheme or (by default) a more sophisticated 4th order Runge-Kutta time scheme. The Runge-Kutta scheme is based on that of *Fisher et al.* [1993]. The switch `ITRAJ` in the subroutine `EVOL` determines which timestepping scheme is used.

6 Physics

6.1 Convection

The convection scheme implemented in TOMCAT/SLIMCAT is based on the work of *Tiedtke* [1989]. It is a mass flux scheme which explicitly represents the cloud fields and their circulation in order to determine the effect of convection on the large-scale budgets of heat and moisture. Representing this circulation is vital for tracer transport because the convective-induced fluxes are needed. Other schemes, such as the convective adjustment scheme of *Betts and Miller* [1984] and the *Kuo* [1974] scheme do not explicitly represent this circulation. By using a simple bulk model to represent the cloud ensemble, the contribution of convection to the large-scale budget equations of heat and moisture can be calculated in the Tiedtke scheme as:

$$\begin{aligned}\frac{\partial \bar{s}}{\partial t} + \mathbf{v}\cdot\nabla\bar{s} + \bar{w}\frac{\partial \bar{s}}{\partial t} = & - \frac{1}{\bar{\rho}_{air}}\frac{\partial}{\partial z}[M_u s_u + M_d s_d - (M_u + M_d)\bar{s}] + \mathrm{L}(c_u - e_d - \tilde{e}_l - \tilde{e}_p) \\ & - \frac{1}{\bar{\rho}_{air}}\frac{\partial}{\partial z}(\bar{\rho}_{air}\overline{w's'})_{tu} + \bar{Q}_R \qquad (17)\end{aligned}$$

$$\begin{aligned}\frac{\partial \bar{q}}{\partial t} + \mathbf{v}\cdot\nabla\bar{q} + \bar{w}\frac{\partial \bar{q}}{\partial t} = & - \frac{1}{\bar{\rho}_{air}}\frac{\partial}{\partial z}[M_u q_u + M_d q_d - (M_u + M_d)\bar{q}] + \mathrm{L}(c_u - e_d - \tilde{e}_l - \tilde{e}_p) \\ & - \frac{1}{\bar{\rho}_{air}}\frac{\partial}{\partial z}(\bar{\rho}_{air}\overline{w'q'})_{tu} \qquad (18)\end{aligned}$$

where s is the dry static energy, q the specific humidity, ρ_{air} the density of air, $\mathbf{v}$ the horizontal velocity vector, w the vertical velocity, c the rate of condensation, e the rate of evaporation and Q_R the radiative heating. Overbars denote grid area means, primes denote deviations from the mean. $\tilde{e}_l$ is the evaporation of cloud air that has been detrained into the environment and $\tilde{e}_p$ is the evaporation of precipitation in the unsaturated subcloud layer. The index tu denotes boundary layer turbulence. For both expressions the first two terms on the right side represent the perturbation to the heat and moisture budgets by convection.

The convection scheme implemented in TOMCAT/SLIMCAT is identical to that described by Tiedtke [1989] apart from the following differences: in TOMCAT midlevel convection and convective downdrafts are not included, and there is no organized entrainment of environmental air above cloud base. The scheme does include cumulus updrafts in the vertical column, entrainment of environmental air into the cloud and detrainment of cloud air to the environment. The magnitudes of these are related to horizontal convergence of moisture below cloud and the difference between cloud and environmental specific humidity at cloud base. Mass balance within the vertical column is maintained by including sub-grid subsidence of environmental air, (induced by convection), within the same timestep. Subsidence is parameterized differently to *Tiedtke* [1989].

6.1.1 Convective Updraft

The $\mathcal{N}$ vertical levels in TOMCAT extend from level $\mathcal{N}$, closest to the Earth's surface, to level 1 (e.g. at 10 hPa in the stratosphere with old ECMWF grid). Thermodynamic and other variables are defined either at the centre of model levels or at level interfaces, (see Figure 1).

Air at the top of level $\mathcal{N}$ (i.e. at interface $\mathcal{N}-1$) is lifted through model level $(\mathcal{N}-1)$ along the dry adiabat:

$$\frac{dT_p}{dz} = -\frac{g}{c_p}$$

(The notation used in this report is summarised in Appendix 2).

The temperature of the lifted parcel at the top of level $(\mathcal{N}-1)$ (i.e. interface $\mathcal{N}-2$) is given by

$$T_p^{\mathcal{N}-2} = T_p^{\mathcal{N}-1} - (z^{\mathcal{N}-2} - z^{\mathcal{N}-1})g/c_p$$

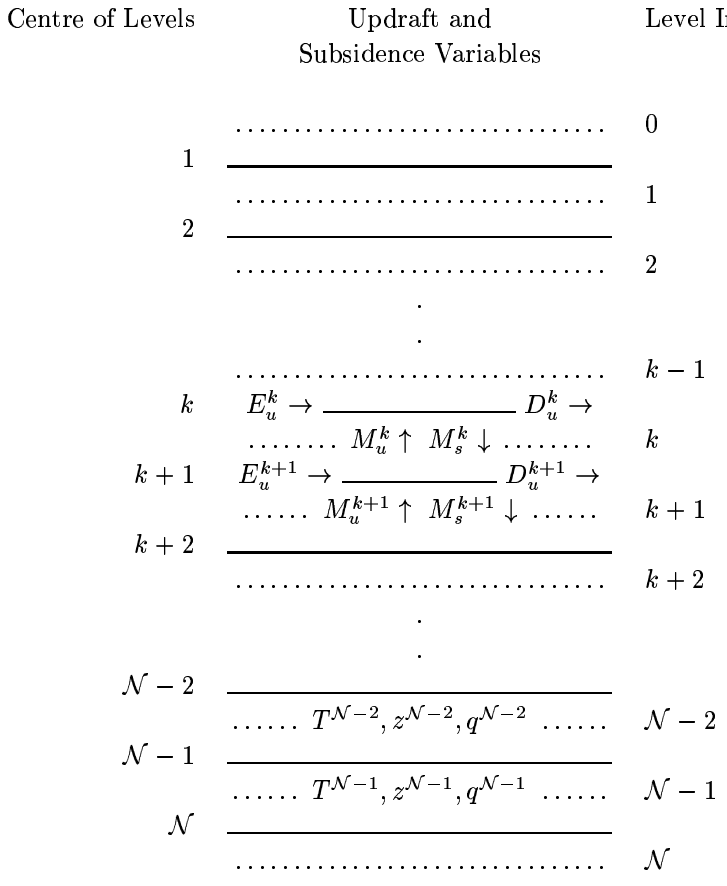


Figure 1: Convections variables shown on TOMCAT/SLIMCAT vertical levels

Although the specific humidity remains constant with parcel lifting, at this higher level the temperature of the lifted parcel is lower than at interface ($\mathcal{N} - 1$) and therefore the relative humidity (RH) increases. If the lifted parcel becomes supersaturated at interface ($\mathcal{N}$ - 2) so that $RH > 100\%$, then the first criterion for cloud occurrence in the column has been met and this interface is set as cloud base. If $RH < 100\%$, the parcel is lifted to the next level, tested for supersaturation, and so on until cloud base is found.

At cloud base, the lifted parcel now lies in a regime of moist adiabatic ascent. Excess moisture is condensed and the specific humidity q_p, and temperature T_p, are iteratively adjusted to their saturation ($RH = 100\%$) values q_p', T_p', using

$$q_p' = q_p - \frac{q_p - q_{sat}(T_p)}{1 + \frac{L}{c_p}\frac{dq_{sat}(T_p)}{dT}}$$

$$T_p' = T_p - (q_p' - q_p)\frac{L}{c_p}$$

from

$$\triangle T = -\triangle q \frac{L}{c_p}, \qquad \triangle q = q' - q$$

and

$$q' \simeq q_{sat}(T) + \frac{dq_{sat}}{dT} \triangle T$$

To achieve convergence towards saturation values, in each subsequent iteration the new values of q'_p and T'_p replace q_p and T_p. Finally, cloud liquid water content is set equal to the total condensate.

The lifted parcel is now tested for buoyancy with respect to the environmental air[2]. If the virtual static energy of the parcel, s_{vp}, is less than that of the environment, s_{ve}, then the parcel is not buoyant and the model vertical column is set as non-convecting.

$$\begin{aligned} s_{vp} &= s_p + c_p\, T_p\, (0.608\, q_p - LWC) \\ s_{ve} &= s_e + c_p\, T_e\, (0.608\, q_e) \end{aligned}$$

where

$$\begin{aligned} s_p &= \mathrm{c}_p\, T_p + \mathrm{g\ z} \\ s_e &= \mathrm{c}_p\, T_e + \mathrm{g\ z} \end{aligned}$$

If $s_{vp} > s_{ve}$ the parcel is buoyant and the second criterion for convective cloud occurrence within the column is met. The final criterion is the horizontal convergence of moisture below cloud base. This is calculated using:

$$-\int_{\text{surface}}^{\text{cloud base}} \{\mathbf{v}.\nabla\rho_w\}\, dz = \int_{\text{surface}}^{\text{cloud base}} \{\rho_w\ \nabla.\mathbf{v} - \nabla(\rho_w\ \mathbf{v})\}\, dz$$

If this integral is positive the grid column is convectively unstable and **deep convection** occurs. However, if there is a divergence of moisture, a further check is made to determine whether the column is unstable with respect to **shallow convection**. Shallow convection depends less on large-scale moisture convergence than on evaporation of moisture from the surface. If addition of this local surface evaporative flux to the integral above results in a positive water vapour flux at cloud base, then shallow convection occurs within the column. If there is still a divergence of moisture below cloud base, the model column is set as cloudless.

To maintain moisture content during convection, a moisture balance is imposed below cloud base. The updraft mass flux of air through cloud base is therefore calculated (for shallow and deep convection) as:

$$M_u^{base} = \frac{-\int_{\text{surface}}^{\text{cloud base}} \mathbf{v}.\nabla\rho_w dz\ +\ \text{surface evaporation}}{(q_p - q_e)_{\text{at cloud base}}}$$

and the updraft cloud base mass flux of tracer Tr_u^{base} as;

$$Tr_u^{base} = M_u^{base}\ \chi_p^{tr}\ .$$

The updraft mass flux at cloud base is generated by organized entrainment of air from levels below cloud base. The fraction originating from each of these levels is set proportional to the level mass.

The lifted parcel is raised through the next model level. During this process, turbulent entrainment into (E_u) and detrainment out of (D_u) the cloud are parameterized as:

$$E_u = M_u\ \epsilon_u\ \ ; \qquad D_u = M_u\ \delta_u$$

where the fractional entrainment/detrainment rates (ϵ_u, δ_u) depend inversely on cloud radii. Clouds of smaller sizes are assumed to occur in the absence of large-scale convergence (i.e. shallow convection), and it is further assumed that $\epsilon_u = \delta_u$, so that a simple form is used:

$$\epsilon_u = \delta_u = \left\{ \begin{array}{ll} 1\mathrm{x}10^{-4}\ m^{-1}, & \text{for deep convection} \\ 3\mathrm{x}10^{-4}\ m^{-1}, & \text{for shallow convection} \end{array} \right\}$$

[2]No test is made below cloud base for parcel buoyancy, because it is assumed that large scale moisture convergence, (which is the final boundary condition for cloud occurrence), sufficiently destabilizes the below cloud levels and ensures thermals reach the lifting condensation level, LCL. (Note that if buoyancy is not achieved at the LCL, in this parameterization scheme there is no further lifting of the parcel to test for a higher level of free convection).

The updraft mass flux and tracer mass flux vary as:

$$\begin{aligned} \frac{\partial M_u}{\partial z} &= E_u - D_u \\ \frac{\partial Tr_u}{\partial z} &= E_u\ \chi_e - D_u\ \chi_p = E_u\ \chi_e - D_u\ \frac{Tr_u}{M_u} \end{aligned}$$

During ascent through the level, T_p and q_p are adjusted along the wet adiabat and modified for entrainment of environmental air and detrainment of cloud air.[3] Any condensate is added to the liquid water loading. Detrainment of liquid water to the environment is included. At the level top, the lifted parcel is tested for buoyancy as before; if it is buoyant then lifting through subsequent levels continues, otherwise the lower interface of the model level in which buoyancy is lost is set as the top of the cloud column, k_{top}, (see Figure 2).

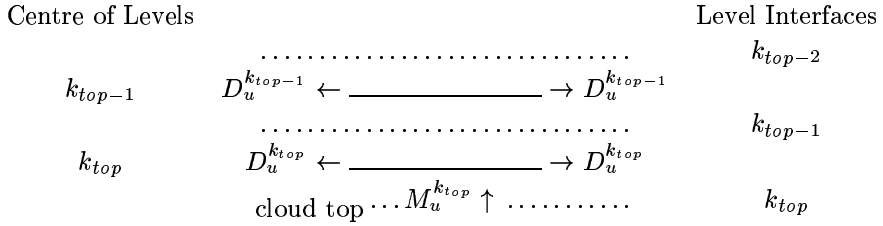


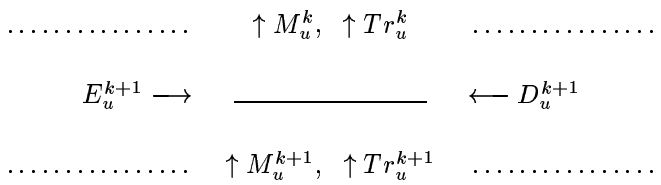
Figure 2: Showing cloud top and detrainment of cloud air

Organized detrainment occurs above cloud top[4] and allows for some small shallow cumuli overshooting the level of zero buoyancy [Tiedtke, 1989]. It is parameterized as

$$\begin{aligned} D_u^{k_{top}} &= (1-\beta)\ M_u^{k_{top}-1} \\ D_u^{k_{top}-1} &= \beta\ M_u^{k_{top}-1} \end{aligned}$$

$$\beta = \begin{cases} 0.3 & \text{for shallow convection,} \\ 0.0 & \text{for deep convection.} \end{cases}$$

Discretization



At a model level k

$$M_u^k - M_u^{k+1} = E_u^{k+1} - D_u^{k+1}\ .$$

The tracer updraft mass flux can be implicitly calculated as;

$$Tr_u^k - Tr_u^{k+1} = E_u^{k+1}\ \chi_e^{k+1} - D_u^{k+1}\ \frac{Tr_u^k}{M_u^k}.$$

[3]No modification is made to the large scale (environmental) thermodynamic properties (q_e, T_e), even though a change in these would be expected due to detrainment of cloud air.

[4]Note that there is turbulent entrainment into and detrainment out of level k_{top} in addition to organized detrainment.

Substituting $M_u^k = M_u^{k+1} + E_u^{k+1} - D_u^{k+1}$, and $\chi_e^{k+1} = S0^{k+1}/SM^{k+1}$, and solving for Tr_u^k gives;

$$Tr_u^k = (Tr_u^{k+1} + E_u^{k+1}\frac{S0^{k+1}}{SM^{k+1}})(1 - \frac{D_u^{k+1}}{M_u^{k+1} + E_u^{k+1}}). \tag{19}$$

The updraft tracer flux through interface k can be represented as a linear function

$$Tr_u^k = \sum_{j=\mathcal{N}}^{1} S0^j \ \Phi_{k,j}^u$$

where $\Phi_{k,j}^u$ **is the fractional tracer mass from level j transferred through interface k in the convective updraft**. From the condition that, at the lowest model interface $\mathcal{N}$, $M_u^{\mathcal{N}} = Tr_u^{\mathcal{N}} = 0$, it can be seen that

$$\Phi_{\mathcal{N},j}^u = 0 \qquad j = 1, 2, \ldots, \mathcal{N},$$

i.e. there is no flux through the lowest model interface. From equation (19) a recursion formula can be used to determine general column values of $\Phi_{k,j}^u$:

$$\Phi_{k,j}^u = (\Phi_{k+1,j}^u + \delta_{k,j-1}\frac{E_u^{k+1}}{SM^{k+1}})(1 - \frac{D_u^{k+1}}{M_u^{k+1} + E_u^{k+1}})$$

where

$$\delta_{k,j-1} = \begin{cases} 1 & \text{if } j = k+1 \text{ (adjacent levels),} \\ 0 & \text{otherwise.} \end{cases}$$

6.1.2 Subsidence

Outside of the cloud a sub-gridscale subsidence flux is induced to maintain mass balance within the column. The mass flux through interface k, M_s^k is given by

$$M_s^k = -M_u^k$$

so that the tracer flux is $Tr_s^k = -M_u^k \ \chi_e^k$, and $\Phi_{k,j}^u = -\frac{M_u^k}{SM^k}\delta_{k,j}$.

6.2 PBL

Louis [1979] or Holtslag and Boville [1983]

6.3 Radiation

Within SLIMCAT a radiation scheme is used to diagnose heating rates to calculate the vertical motion in the isentropic part of the model. There are different options for the radiation scheme.

Details of CTM Radiation Schemes

	MIDRAD	CCMRAD	ZHMRAD
Temperature	Analyses	Analyses	Analyses
Domain	0 - 700 hPa	0 hPa - surface	0 hPa - surface
Longwave	CO_2 15 μm	O_3, CO_2, H_2O,	
	O_3 9.6 μm	CH_4, N_2O, F11, F12	
	vibro bands H_2O	0 - 3000 cm^{-1}	
		$\Delta\lambda$ 100 cm^{-1}	
Shortwave	O_2 and O_3	O_2, O_3,	
	6 bands	CO_2, H_2O	
	125-175, 175-205 nm,	18 intervals	
	206-244, 244-278 nm,	200 - 5000 nm	
	278-363, 408-853 nm,		
Albedo	0.2 $\lambda_w > 278$ nm	Climatology	
		fn(λ_w, t, ϕ, λ_w)	
Clouds	None	Not included	
Refs	*Shine* [1987]	*Briegleb* [1992a, b]	
	Shine and Rickaby [1989]		

7 Chemistry Schemes

A range of different 'chemical' schemes can be included in TOMCAT/SLIMCAT. These can be standard 'full' chemistry schemes for the stratosphere or troposphere, or can be simple schemes written/supplied by the user. In all cases the basic approach is that the TOMCAT/SLIMCAT job has a chemistry subroutine (called `CHIMIE`). If this subroutine is short and does not modify the model tracers then they are 'passive'. On the other hand the `CHIMIE` subroutine can call the subroutines for a full chemistry scheme which may include emissions, deposition etc, as well as the integration of the chemistry itself.

7.1 Stratospheric

There is a standard stratospheric chemistry scheme contained in the TOMCAT/SLIMCAT library (in the nupdate deck `TOMCHI`).

See separate manual (Volume II) for details.

7.2 ASAD-based 'Tropospheric'

The 'full' tropospheric chemistry scheme is based on the ASAD solver [*Carver et al.*, 1997] (which is contained in its own nupdate library `UNIASADx.x`). The standard code for the default scheme is contained in the nupdate deck `TROPCHI`.

7.3 Cariolle Ozone

[[parameterised O3]]

7.4 User Supplied

The user can supply his/her own version of the `CHIMIE` subroutine to modify the advected tracers to suit any particular experiment. An example of a subroutine for a single 'age tracer' is given here.

```
      SUBROUTINE CHIMIE
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C        DUMMY INTERFACE SUBROUTINE FOR CHEMISTRY
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IMPLICIT NONE
*CALL,PARADI
*CALL,CSTUNI
*CALL,FOR3D
*CALL,REDEM
*CALL,GRILLE
*CALL,MOMENT
C
      REAL FAC
      INTEGER I,IAN,IBIS,JOUR,JV,K,L,MOIS,NUMJOU
C
C
C     Work out day of year
      JOUR =INT(RJOUR)
      MOIS =INT(RMOIS)
```

```
      IAN  =INT(RAN)
      CALL WCALEN(NUMJOU,IBIS,JOUR,MOIS,IAN)
C
C     Overwrite bottom level with current time
      DO 40 K=1,LAT
        DO I=1,LON
          SO(I,K,NIV,1)=(RAN + (RMOIS-1.0)/12.0 + RJOUR/365.0)
     1                  *SM(I,K,NIV)
        ENDDO
 40   CONTINUE
C
C     Store advected tracers (SO) in output arrays (ST)
      DO 50 L=1,NIV
        DO K=1,LAT
        DO I=1,LON
          ST(I,K,L,1 )=SO(I,K,L,1)
        ENDDO
        ENDDO
 50   CONTINUE
C
      RETURN
      END
```

7.5 GLOMAP

GLOMAP is a set of subroutines which forms a module for calculating aerosols properties (e.g. see Mann, 2004). This can be included in the TOMCAT/SLIMCAT model. In this case the `CHIMIE` subroutine calls the GLOMAP aerosol subroutines (as well as the ASAD chemical routines).

7.6 DLAPSE

DLAPSE is a lagrangian particle sedimentation scheme used to calculate denitrification in the polar stratosphere (e.g. Davies, 2004). Originally this code was a separate IDL model, but it is now inserted into the TOMCAT/SLIMCAT trajectory subroutines using F90.

8 Forcing Winds and Temperatures

This section describes the various sources of winds and temperatures that can be used to force TOMCAT/SLIMCAT.

8.1 ECMWF Analyses

The ECMWF analyses that can be used to force TOMCAT are stored on HPCx in the directories `/hpcx/home/n02/n02/lrkd/data/ECMWF`. A subset of these are in Leeds on `/nfs/portsdown_portsdown02/martyn/ECMWF`. The files contain the spectral coefficients of the streamfunction, velocity potential, temperature and specific humidity on the original ECMWF model levels (i.e. 60 levels) as well as the surface pressure. The table lists the periods that are currently available:

Type	Period	Resolution
ERA-40	1/1/1957 to 31/12/2002	T42
Operational	1/11/1999 to 29/2/92	T42

8.2 UK Met. Office UARS Analyses

TOMCAT/SLIMCAT can also be run using the gridpoint analyses of the U.K. Met. Office produced for the UARS mission. However, this is not recommended for certain configurations of the model as the calculation of vertical motion from the divergence can lead to noisy fields. These analyses are produced on 22 isobaric levels from 0.3hPa down to 1000hPa. The analyses are produced on an Arakawa 'B' grid so that the wind fields (u and v) are not given at the same location as temperature and geopotential. TOMCAT/SLIMCAT reads the data on this staggered grid and the fields are interpolated within the model. This data is stored in `/hpcx/home/n02/n02/lrkd/data/UKMO` and also `/nfs/portsdown_portsdown02/martyn/UKMO`. The table lists the periods that are currently available:

Period	Resolution
21/10/1991 to 31/10/2003	2.5° lat. x 3.75° lon.

9 Running the TOMCAT/SLIMCAT Model

9.1 Parameters

The table lists some of the main parameters in TOMCAT.

Parameter	Meaning
LAT	number of latitudes
LON	number of longitudes
MI0	spectral truncation of winds after conversion
MI1	spectral truncation of winds read in
MLAT	number of latitudes in gridpoint forcing file
MLON	number of longitudes in gridpoint forcing file
NIV	number of levels

9.2 Model Switches and Variables

The following switches are contained in the common deck SWITCH and can be set in the model jobdeck:

Switch	Meaning	Values
ICAT	Vertical grid	1 = SLIMCAT (sigma-theta) 2 = TOMCAT (sigma-pressure)
IVER	Vertical motion	0 = None 1 = Divergence 2 = Mixed Divergence/heating rates 3 = Heating rates
ICONV	Convection	0 = none 1 = Tiedtke scheme
IVDIF	Vertical mixing	0 = none 1 = Louis scheme 2 = Simple complete mixing 3 = Holtslag and Boville
IDRY	Dry deposition	0 = none 1 = Old scheme 2 = New scheme (needs IVDIF=3)
IWET	Wet deposition	0 = none 1 = Giannakopolous Scheme 1 2 = Giannakopolous Scheme 2 3 = Giannakopolous Scheme 3

The length of the TOMCAT model run is controlled by the variables read in on channel 94

```
cat <<'eof'> fort.94
0     NO OF FILES TO JUMP IN FORCING FILE
0     =0 INITIALLY, =1 RESTART
40    NCYCLT (NDAYS * NO FORCING FILES/DAY)
0     NFFILE (NO FORCING OUTPUTS PER FILE)
eof
```

The first line and last line can be ignored. The second line indicates whether the start data comes from an initial data file (=0) or from a restart file (=1). The third line specifies the length of the run in numbers of analysis times.

```
#
#  file at forcing resolution
cp $MARTYN/UTIL/TRONxx  fort.2
#  files at model grid resolution
cp $MARTYN/UTIL/TRONyy  fort.1
cp $MARTYN/UTIL/LEGCyy  fort.20
cp $MARTYN/UTIL/LEGIyy  fort.21
#
```

The above data files need to be correctly set according to the values of MI0 and MI1. xx=MI1 and yy=MI0.

The model outputs a number files. On channel 25 the model writes a PDG file which contains the 3D tracer arrays at intervals specified by the variable `NSO1`. On channel 31 the model outputs REST file which can be used to restart a continuation run.

```
#
# jour=n results
mv fort.25 $EXP/UNI001.PDG01
mv fort.12 $EXP/UNI001.DIAG1
mv fort.15 $EXP/UNI001.ZON01
mv fort.31 $EXP/UNI001.REST1
mv fort.14 $EXP/UNI001.TRAJ1
#
```

The following lines in the subroutine `INIEXP` control the length of the model timestep and frequency of output.

```
      DT0=3600.0                                              INIEXP.56
      NITERT=24                                               INIEXP.57
      NDYN=1                                                  INIEXP.58
      NSO1=24                                                 INIEXP.59
      NSO2=24                                                 INIEXP.60
      NSO3=24                                                 INIEXP.61
      NSO4=24                                                 INIEXP.62
      NSO5=1                                                  INIEXP.63
```

`DT0` is the basic model timestep. This is split into `NDYN` dynamical subtimesteps. `NITERT` is the number of iterations in one cycle, i.e. the time between the forcing files divided by `DT0`. Output is written to the PDG file every `NSO1` iterations, to the DIAG file every `NSO2` iterations, to the STAT file every `NSO3` iterations, and to the TRAJ file every `NSO4` iterations. `NSO5` is a switch to write to the ZON files (=1) or not (=0).

The Prather advection scheme can use a limiter to prevent negative mixing ratios, as described by *Prather* [1986]. This limiter can be switched on by setting the variable `LIMIT` to `TRUE`. Note that the use of this limiter can destroy tracer correlations (as the advection is no longer independent of the tracer distribution) and so should be used with care.

```
      LIMIT=.FALSE.                                           ADVEC.13
```

The following table lists some of the main model variables and the Fortran common decks in which they are stored.

Variable	Common Deck	Meaning	Units
FLT	GRILLE	defines interlevel pressure	none
GLT	GRILLE	defines interlevel pressure	none
PL	FOR3D	centrelevel pressure	Pa
PLT	FOR3D	interlevel pressure	Pa
Q3D	FOR3D	specific humidity at box centre	kg/kg
SM	MOMENTS	total mass of box	kg
S0	MOMENTS	zero order moment	kg*vmr
SX	MOMENTS	fist order moment in x direction	kg*vmr
SXX	MOMENTS	second order moment in x direction	kg*vmr
SURF	GRILLE	surface area of grid cell	m^2
T3D	FOR3D	temperature at box centre	K
U3D	FOR3D	E-W wind (x direction) at box centre	ms^{-1}
V3D	FOR3D	N-S wind (y direction) at box centre	ms^{-1}
W3D	FOR3D	Pressure coordinate vertical velocity	Pa s^{-1}

9.3 Fortran Channels

The models makes use of the following Fortran channels during execution:

Channel	Variable	Common Deck	Purpose
1	ITAB		Read file TRONyy
2	ITAB1		Read file TRONxx
9	IFSO1	REDEM	Write PDG file
12	IFSO2	REDEM	Write DIAG file
13	IFSO3	REDEM	Write STAT file
14	IFSO4	REDEM	Write TRAJ file
15	IFSO5	REDEM	Write ZON file
17	ITRIO	REDEM	General I/O channel
18	IEVAP	REDEM	Read file EVAPxx
19	ICON	REDEM	Writing/reading convection/diffusion matrix
20	IFPP0		Read file LEGCyy
21	IFPP1		Read file LEGIyy
30	IFRD	REDEM	Read restart file
31	IFRF	REDEM	Write restart file
32	IFRT	REDEM	Trajectory restart file
40	IFORC	REDEM	Read forcing files
94			Read information about length of run etc.

9.4 Subroutines

The following diagram shows the structure of the model. The cycle loop corresponds to the frequency of the forcing winds and temperatures (typically 6 or 24 hours). The iteration loop corresponds to the basic model timestep (set by DT0).

- ADVEC Calls the subroutines to perform advection.
- CHIMIE Interface between the chemical model and the dynamical model.
- CALSUB Sets up convection and vertical diffusion terms.

- CONVEC Applies convection and vertical diffusion.
- FINCYCL End of cycle. Writes REST file.
- FINITER End of iteration. Writes PDG, DIAG, ZON files.
- INICYCL Start of cycle (period of forcing analyses).
- INIEXP Initialise experiment.
- INITER Start of iteration (DT0 timestep).

9.5 Advection

The choice of advection scheme is set using the parameter IADV:

```
PARAMETER(IADV=3)                                    SWITCH.23
```

IRAD	Scheme	Name	Comment
0	none		
1	MIDRAD	Midrad	
2	CCMRAD	CCM	
3	SLT	ZH Morcrette	

9.5.1 Prather Scheme

```
LIMIT=.FALSE.                                        ADSLT.20
```

9.5.2 SLT Scheme

A switch to ensure interpolated tracer volume mixing ratios are within the max/min limits of the original values:

```
LIMIT=.FALSE.                                        ADSLT.20
```

The latitude at which the coordinate transformation occurs is set by:

```
DLATLIM=45./CONV                                     BACKTRA.37
```

The vertical interpolation can be done in a cubic (arg=1) or linear (arg=0) by specifying xx in the call to POIDZ.

```
CALL POIDZ(Z0,LG0,ALFZM1,ALFZP0,ALFZP1,ALFZP2,LPR0,arg) INTER.56
```

Variable	Value	Comment
LIMIT	FALSE	To preserve tracer sums
DLATLIM	45./CONV	
arg	x	xxx interpolation

Recommended and default parameters for SLT scheme

9.6 Output Files

The model produces a range of output files and brief details are given here. Usually the user ftps the output files from the remote machine which runs TOMCAT/SLIMCAT to his/her local machine for analysis and storage. Most of the output files are single precision (32 bit) unformatted binary files. Reading them on some machines (i.e. linux) will require software that switch from big-endian to little-endian binary format.

I have a selection of fortran and IDL programs to read these files. I also have jobs which convert the output (e.g. PDG files) to NetCDF files.

9.6.1 PDG

The PDG file is the basic output from the model run. It contains the global 3D fields of tracers, temperature, PV, humidity and pressure at a requested frequency (controlled by `NSO1`).

9.6.2 DIAG

This file contains basic information on the model transport terms (e.g. mass fluxes, winds etc). Mainly used for debugging/diagnosing the model. (Controlled by `NSO2`).

9.6.3 STAT

This file contains profiles of model tracers, temperature, pressure and PV interpolated to a specified set of stations (currently set to 55). This allows frequent sampling of the model at just a few sites. (Controlled by `NSO3`).

9.6.4 ZON

This file contains zonal mean output on the 15th day of each month. This is a smaller filer than the PDG file and gives a more regular sampling for multiannual runs.

9.6.5 OXO

9.6.6 REST

This file is for restarting a new TOMCAT/SLIMCAT run from the final time of the previous run.

10 Acknowledgements

I am grateful to Pascal Simon of the CNRM, Toulouse for many helpful discussions. [[Others..]]

References

[1] Andrews, D.G., J.R. Holton, and C.B. Leovy, Middle Atmosphere Dynamics, Academic Press, 1987.

[2] Bacmeister, J.T., M.R. Schoeberl, M.E. Summers, J.R. Rosenfield, and X. Zhu, Descent of long-lived trace gases in the winter polar vortex, *J. Geophys. Res.*, **100**, 11,669-11,684, 1995.

[3] Betts, A.K., and M.J. Miller, A new convective adjustment scheme, *ECMWF Technical Report No. 43*, 1984.

[4] Briegleb, B.P., Delta-Eddington Approximation for Solar Radiation in the NCAR Community Climate Model, *J. Geophys. Res.*, **97**, 7603-7612, 1992.

[5] Carver, G.D., P.D. Brown, and O. Wild, The ASAD atmospheric chemistry integration package and chemical reaction database, *Computer Physics Communications*, **105**, 197, 1997.

[6] Chipperfield, M.P., D. Cariolle, P. Simon, R. Ramaroson and D.J. Lary, A three-dimensional modelling study of trace species in the Arctic lower stratosphere during winter 1989-90, *J. Geophys. Res.*, **98**, 7199-7218, 1993.

[7] Chipperfield, M.P., M.L. Santee, L. Froidevaux, G.L. Manney, W.G. Read, J.W. Waters, A.E. Roche, and J.M. Russell, Analysis of UARS data in the southern polar vortex in September 1992 using a chemical transport model, *J. Geophys. Res.*, **101**, 18,861-18,881, 1996.

[8] Chipperfield, M.P., J. Kettleborough and A. Pardaens, The TOPCAT offline trajectory model, UGAMP internal report no. 37, November 1995.

[9] Fisher, M., A. O'Neill, R. Sutton, Rapid descent of Mesospheric air into the Stratospheric Polar Vortex, *Geophys. Res. Lett.*, **20**, 1267-1270, 1993.

[10] Giannakopoulos, C., M.P. Chipperfield, K.S. Law, and J.A. Pyle, Validation and intercomparison of wet and dry deposition schemes using Pb-210 in a global three-dimensional off-line chemical transport model, *J. Geophys. Res.*, **104**, 23,761-23,784, 1999.

[11] Heimann, M., The global atmospheric tracer model TM2, Technical Report no. 10, *Deutsches Klimarechenzentrum, Hamburg*, 1995.

[12] Holtslag, A.A.M., and B. Boville, Local versus nonlocal boundary layer diffusion in a global climate model. *J. Clim.*, **6**, 1825-1842, 1993.

[13] Kuo, H.L., Further studies of the parameterization of the influence of cumulus convection of large scale flow, *J. Atmos. Sci.*, **31**, 1232-1240, 1974.

[14] Law, K.S., P.-H. Plantevin, D.E. Shallcross, H.L. Rogers, J.A. Pyle, C. Grouhel, V. Thouret and A. Marenco, Evaluation of modeled O_3 using Measurements of Ozone by Airbus In-Service Aircraft (MOZAIC) data, *J. Geophys. Res.*, **103**, 25,721-25,737, 1998.

[15] Louis, J.-F., A parametric model of vertical eddy fluxes in the atmosphere, *Boundary Layer Meteorol.*, **17**, 187-202, 1979.

[16] Mann, GLOMAP, 2004.

[17] Prather, M. J., Numerical advection by conservation of second-order moments, *J. Geophys. Res.*, **91**, 6671-6681, 1986.

[18] Prather, M. J., M. McElroy, S. Wofsy, G. Russell and D. Rind, Chemistry of the global troposphere, fluorocarbons as tracers of air motion, *J. Geophys. Res.*, **92**, 6579-6613, 1987.

[19] Rood et al, 1991.

[20] Russell, G.L., and J.A. Lerner, A new finite differencing scheme for the tracer transport equation, *J. Appl. Meteorol.*, **20**, 1483-1498, 1981.

[21] Shine, K.P., The middle atmosphere in the absence of dynamical heat fluxes, *Q. J. R. Meteorol. Soc.*, **113**, 603-633, 1987.

[22] Shine, K.P., Zonal momentum in the middle atmosphere, *Q. J. R. Meteorol. Soc.,* **115,** 265–292, 1989

[23] Shine, K.P., and J.A. Rickaby, Solar radiative heating due to absorption by ozone, in *Proceedings of Quadrennial Ozone Symposium, Gottingen, pp 597-600, A. Deepak, Hampton, Va.*, 1989.

[24] Stockwell, D.Z., and M.P. Chipperfield, A tropospheric chemical transport model: Development and validation of the model transport schemes. *Q. J. Roy. Met. Soc.*, **125**, 1747-1783, 1999.

[25] Stockwell, D.Z., C. Giannakopoulos, P.H. Plantevin, G.D. Carver, M.P. Chipperfield, K.S. Law, J.A. Pyle, D.E. Shallcross, K.Y. Wang, Modelling NOx from lightning and its impact on global chemical fields *Atmospheric Environment*, **33**, No.27, 4477-4493, 1999.

[26] Swinbank, R., and A. O'Neill, A stratosphere-troposphere data assimilation system, *Mon. Weather Rev.*, **122**, 686-702, 1994.

[27] Tiedtke, M., A comprehensive mass flux scheme for cumulus parameterization in large-scale models, *Mon. Wea. Review*, **117**, 1779-1800, 1989.

[28] Williamson, D.L., and P.J. Rasch, Two-dimensional semi-Lagrangian transport with shape preserving interpolation, *Monthly Weather Review*, **117**, 102-129, 1989.

11 Appendix 1. Notation

Symbol	Meaning	Value
A, B	Parameters to define $\sigma - p$ levels	
a	radius of the earth	6.371x10^6 m
C	Parameter to define $\sigma - \theta$ levels	
c_p	heat capacity of dry air at constant pressure	= 1005.46 J/K/kg
D	divergence	
D_m	Mass flux divergence	kg m^{-2}s^{-1}
$f_h(Ri)$	Stability function	adimensional
g	Gravitational acceleration	m s^{-2}
K	Constant for precipitation parameterization	= 2.0 x 10^{-3}
kas	Asymptotic mixing length	= 438.18 m
K_z	Vertical diffusion coefficient	m^2 s^{-1}
l	Mixing length	m
L	Latent heat of condensation at 0^{o}C	J kg^{-1}
LWC	Liquid water content	kg(water)/kg(air)
$M_{k,j}$	Fraction of tracer mass from level j transferred through model interface k due to all sub-gridscale processes	
M^k	Mass flux of air through interface k from updraft, subsidence or vertical diffusion	kg(air) m^{-2} s^{-1}
p	pressure	
p_s	surface pressure	
p_{top}	Pressure at top model interface	Pa
p_θ	Pressure at top of θ level	Pa
p_0	Reference pressure	100000 Pa
Q	Diabatic heating rate	K/s
q	Specific humidity	kg(water)/kg(air)
R	Gas constant	287.05 J/kg
RH	Relative humidity	
Ri	Richardson's number	adimensional
s	Dry static energy	J kg^{-1}
s_v	Virtual static energy	J kg^{-1}
$S0^j$	Tracer mass in level j	kg(tracer)
SM^j	Air mass in level j	kg(air)
t	time	
T	temperature	
T_s	Surface temperature	K
Tr^k	Mass flux of tracer through interface k from updraft, subsidence or vertical diffusion	kg(tracer) m^{-2} s^{-1}
u	zonal wind	
v	meridional wind	
U	u cos(latitude)	
V	v cos(latitude)	
$\vec{\mathbf{v_h}}$	horizontal wind vector	
w	Vertical velocity	Pa/s
w_m	Vertical mass flux	kg m^{-2}s^{-1}
w_m0	Vertical mass flux at top interface	kg/s
z	Geopotential height	m
Δz	Vertical depth of the model level	m

Symbol	Meaning	Value
κ	Ratio R/C_p	
χ^{tr}	Tracer mass mixing ratio	kg/kg(air)
ϕ	latitude	
λ	longitude	
λ	Von Karman's constant	= 0.4 m
λ_w	Wavelength	m
μ	$= sin\phi$	
$\Phi_{k,j}$	Fraction of tracer mass from level j transferred through interface k from updraft, subsidence or vertical diffusion	
ρ	Density	kg m^{-3}
ρ_w	Water density	kg(water) m^{-3}
ρ_{tr}	Tracer density	kg(tracer) m^{-3}
ρ_{air}	Air density	kg(air) m^{-3}
ζ	relative vorticity	
η	vertical coordinate	
θ	Potential temperature	K
θ_0	Reference θ	e.g. 350 K
ω	pressure coordinate vertical velocity	
	Superscripts and Subscripts	
k	Model level k	

12 Appendix 2. Universal Constants

The following table lists the variable names of the universal constants in TOMCAT/SLIMCAT. They are contained in the `fortran` common block `CSTES` which is in the `nupdate` common deck `CSTUNI`.

Variable	Description	
CONV	180/XPI	Value
CP	Specific heat capacity of dry air at constant p	1005.46 $JK^{-1}kg^{-1}$
CPSG	CP/GG	
CPSL	CP/XL	
CPV	Specific heat capacity of water vapour at constant p	1869.46 $JK^{-1}kg^{-1}$
CPVMCP	CPV - CP	
DEUOMG	2*OMEGA	
ECPH	CPV/(CP-1)	
ETV	RVSRA -1	
ETVQ	1 - RASRV	
GG	Acceleration due to gravity	9.80665 ms^{-2}
GSCP	GG/CP	
GSRA	GG/RA	
OMEGA	Earth's speed of rotation	7.292x10^{-5} rad s^{-1}
RA	gas constant for dry air	287.05 $JK^{-1}kg^{-1}$
RASCP	RA/CP	
RASCP2	RA/(2CP)	
RASL	RA/XL	
RASRV	RA/RV	
RTER	radius of the earth	6371229 m
RTER2	RTER*RTER	
RV	gas constant for water vapour	461.51 $JK^{-1}kg^{-1}$
RVSRA	RV/RA	
STEFAN		5.6697x10^{-8} ms^{-2}
TMERGL		271.23K
TOO	ice melting temperature	273.16K
UNSCP	1/CP	
UNSG	1/GG	
VKARMN	von Karman constant	0.4
XL	Latent heat of condensation at 0°C	2.5008x10^{6} Jkg^{-1}
XLF	XLI -XL	
XLI	latent heat of sublimation	2.83456x10^{6} Jkg^{-1}
XLISCP	XLI/CP	
XLISG	XLI/GG	
XLSCP	XL/CP	
XLSG	XL/GG	
XLSRV	XL/RV	
XPI	π	3.14159..
XPOO	reference pressure	10^5 Pa

13 Appendix 3. Details of the Prather Advection Scheme

After describing the principle of the advection method in the simple case of 2D transport in cartesian coordinates, we will show how it is used in the more general 3D case in spherical coordinates.

13.1 Definitions

Consider a domain D in the cartesian grid Oxy covered by a regular grid of spacing X and Y (in the following discussion the boxes of the grid will be referenced by the indices n in the direction Ox and k in the direction Oy). Considering a portion B of the grid, of which, for simplicity, the lower left hand corner is the origin of the grid (i.e. $B = [0, X] \times [0, Y]$), we can define the local functions on this portion:

$$K_0(x,y) = 1 \tag{20}$$

$$K_x(x,y) = \frac{2}{X}\left(x - \frac{X}{2}\right)$$

$$K_y(x,y) = \frac{2}{Y}\left(y - \frac{Y}{2}\right)$$

$$K_{xx}(x,y) = \frac{6}{X^2}\left(x^2 - Xx + \frac{X^2}{6}\right)$$

$$K_{yy}(x,y) = \frac{6}{Y^2}\left(y^2 - Yy + \frac{Y^2}{6}\right)$$

$$K_{xy}(x,y) = \frac{4}{XY}\left(x - \frac{X}{2}\right)\left(y - \frac{Y}{2}\right)$$

These functions K_α form a basis for the ensemble of the polynomial functions of order less than or equal to 2 over the box B. In addition, this basis is orthogonal because:

$$\int_0^X \int_0^Y K_\alpha(x,y) . K_\beta(x,y)\, dxdy = 0 \tag{21}$$

for all pairs (α, β) in $\{0, x, y, xx, xy, yy\}$ which satisfy $\alpha \neq \beta$. However, the basis is not normalised, and we have:

$$\int_0^X \int_0^Y K_0^2(x,y)\, dxdy = XY = S \tag{22}$$

$$\int_0^X \int_0^Y K_x^2(x,y)\, dxdy = S/3$$

$$\int_0^X \int_0^Y K_y^2(x,y)\, dxdy = S/3$$

$$\int_0^X \int_0^Y K_{xx}^2(x,y)\, dxdy = S/5$$

$$\int_0^X \int_0^Y K_{yy}^2(x,y)\, dxdy = S/5$$

$$\int_0^X \int_0^Y K_{xy}^2(x,y)\, dxdy = S/9$$

The moments of a function $r(x,y)$ over the box B are defined by the following equations:

$$M_0 = \int_0^X \int_0^Y \rho\, K_0(x,y)\, r(x,y)\, dxdy \tag{23}$$

$$M_x = 3\int_0^X \int_0^Y \rho\, K_x(x,y)\, r(x,y)\, dxdy$$

$$M_y = 3\int_0^X \int_0^Y \rho\, K_y(x,y)\, r(x,y)\, dxdy$$

$$M_{xx} = 5\int_0^X \int_0^Y \rho\, K_{xx}(x,y)\, r(x,y)\, dxdy$$

$$M_{yy} = 5\int_0^X \int_0^Y \rho\, K_{yy}(x,y)\, r(x,y)\, dxdy$$

$$M_{xy} = 9\int_0^X \int_0^Y \rho\, K_{xy}(x,y)\, r(x,y)\, dxdy$$

where ρ represents the mass density, taken to be uniform in the box B.

Equally, knowledge of the moment of a function $r(x,y)$ allows us to have an analytic expression of the function, or more exactly of its projection over the basis of K_α:

$$r(x,y) = \frac{1}{M}\sum_\alpha M_\alpha K_\alpha(x,y) \tag{24}$$

where $M = \rho S$ is the total mass of the box B.

Interpretation of the moments

As the functions K_α are dimensionless, if the function $r(x,y)$ is the mass mixing ratio of a constituent, the moments of r as defined above have the dimension of mass. In particular, the zero order moment, M_0, represents the total mass of tracer contained in the box B. The moment of order 1 in a given direction represents the "average" gradient over the box in this direction; the first order moments also give the position, relative to the centre of the box B, of the centre of mass of the tracer whose distribution is represented by $r(x,y)$. The second order moments are proportional to the curve of the tracer distribution, and give the matrix of the moments of inertia of the tracer distribution $r(x,y)$ for the box considered. The variance of the distribution inside the box can equally be determined from the different moments:

$$\int_0^X \int_0^Y \rho\, r(x,y)\, dxdy = M_0 \tag{25}$$

$$\int_0^X \int_0^Y \rho\, r(x,y)^2 dxdy = M_0 + \frac{1}{3}(M_x + M_y) + \frac{1}{5}(M_{xx} + M_{yy}) + \frac{1}{9}M_{xy}$$

An important property, which arises from the above definitions, is that the moments of a function $r(x,y)$ contain the intrinsic properties of the grid in the sense that they remain invariant under all transformation on the coordinate system; in other words, all solid translations and/or stretches (compressions) of the box B do not change the moments.

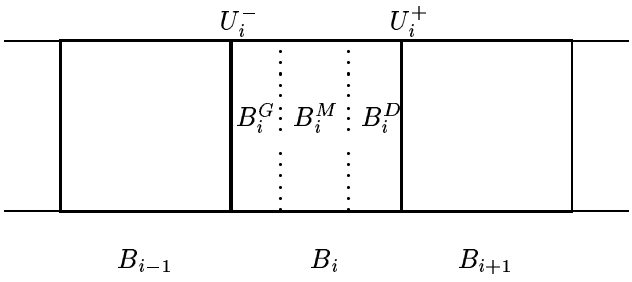


Figure 3: Example one-dimensional grid.

13.2 Principle of scheme

Using the above definitions, the advection of a tracer over the period t to $t + dt$, is performed in several steps:

1. Split each box of the domain into several sub-boxes, as a function of the mass flux at the interfaces between the boxes at the instant t.

   Considering a box B_i (figure 1) and motion along the direction Ox, 3 cases are possible depending on the values of U_i^- and U_i^+, the mass fluxes across the interfaces (B_{i-1}, B_i) and (B_i, B_{i+1}).

   a) U_i^- and U_i^+ are of the same sign

   b) U_i^- positive and U_i^+ negative (convergence)

   c) U_i^- negative and U_i^+ positive (divergence).

   The box B_i is this split into a maximum of 3 sub-boxes:

   - B_i^G contains the air which is going to leave the box B_i by the left face (if U_i^- is negative).
   - B_i^D contains the air which is going to leave the box B_i by the right face (if U_i^+ is positive).
   - B_i^M contains the air which is going to stay in the box B_i during the timestep.

2. Calculate the sub-moments of each sub-box.

   The following equations allow the calculation of the sub-moments of two sub-boxes B^D and B^G of the box B split into two along Ox. For the right sub-box $B^D = [X', X] \times [0, Y]$:

$$M_0^D = \alpha \left[M_0 + (1-\alpha) M_x + (1-\alpha)(1-2\alpha) M_{xx} \right] \tag{26}$$

$$M_x^D = \alpha^2 \left[M_x + 3(1-\alpha) M_{xx} \right]$$

$$M_{xx}^D = \alpha^3 M_{xx}$$

$$M_{xy}^D = \alpha^2 M_{xy}$$

$$M_y^D = \alpha \left[M_y + (1-\alpha) M_{xy} \right]$$

$$M_{yy}^D = \alpha M_{yy}$$

   and for the left sub box $B^G = [0, X'] \times [0, Y]$:

$$M_0^G = (1-\alpha) \left[M_0 - \alpha M_x - \alpha (1-2\alpha) M_{xx} \right] \tag{27}$$

$$M_x^G = (1-\alpha)^2 \left[M_x - 3\alpha M_{xx} \right]$$

$$M_{xx}^G = (1-\alpha)^3 M_{xx}$$
$$M_{xy}^G = (1-\alpha)^2 M_{xy}$$
$$M_y^G = (1-\alpha)\left[M_y - \alpha M_{xy}\right]$$
$$M_{yy}^G = (1-\alpha) M_{yy}$$

where $\alpha = (X - X')/X$ represents the fraction of mass of the box B contained in the box B^D

3. Regroup the different sub-boxes to reconstruct the new boxes at the instant $t + dt$.

   We use the following equations (the reverse of the above) to calculate the moments of a box B from those of the two sub-boxes B^D and B^G which comprise it:

$$M_0 = M_0^G + M_0^D \tag{28}$$
$$M_x = \alpha M_x^D + (1-\alpha) M_x^G + 3\left[(1-\alpha) M_0^D - \alpha M_0^G\right]$$
$$M_{xx} = \alpha^2 M_{xx}^D + (1-\alpha)^2 M_{xx}^G + 5\alpha(1-\alpha)\left[M_x^D - M_x^G\right]$$
$$M_{xx} = +5(1-2\alpha)\left[(1-\alpha) M_0^D - \alpha M_0^G\right]$$
$$M_y = M_j^G + M_y^D$$
$$M_{yy} = M_{yy}^G + M_{yy}^D$$
$$M_{xy} = \alpha M_{xy}^D + (1-\alpha) M_{xy}^G + 3\left[(1-\alpha) M_y^D - \alpha M_y^G\right]$$

   where $\alpha = M_0^D/(M_0^D + M_0^G)$

Reconstruction of a box

When it is necessary to reconstruct a box from three sub-boxes B_1, B_2 and B_3, we proceed by first regrouping B_1 and B_2 (using 19), then adding B_3 to this result. We can verify that the final result is identical to that obtained if we begin by first regrouping B_2 and B_3, and then add B_1. The same result is valid for the splitting of a box into 3 sub-boxes.

Alternate Directions ("Time-splitting")

In a multi-dimensional motion, the method of evaluating the total advection by successively calculating the advection along each direction is known as "time-splitting". This method is not the only one possible, because, in principle, it is possible to proceed in one step by splitting each box into 3N sub-boxes according to the destination of the tracer mass. However, it is is much easier, and less costly, to proceed by separating the directions. In addition, this method gives the possibility of using a different timestep according to the direction, in relation to the different Courant numbers in each direction. Thus, for their experiments with a 3D off- line CTM, Prather et al. [1987] used the following sequence for the advection steps:

- Advection along X (Δt_x)
- Advection along Y (Δt_y)
- Advection along X (Δt_x)
- Advection along Z (Δt_z)
- Advection along X (Δt_x)

- Advection along Y (Δt_y)
- Advection along X (Δt_x)

with $\Delta t_z = 4\Delta t_x$ and $\Delta t_y = 2\Delta t_x$.

The validity of this approach depends on the shear of the wind, rather than on its magnitude. In the case of a uniform wind field, the successive treatment of different directions does not introduce an error with respect to a simultaneous treatment; however, in regions of strong wind shear, to treat directions successively can cause errors in the transport, but it is difficult to estimate the effect quantitatively.

13.3 Generalisation of the Scheme

Case of a variable density

In the above we assumed that the density was constant and uniform over all of the domain; in fact, the equations obtained remain valid even if this is not the case, due to the property described above. Suppose that we had calculated the moments of a box B from those of the two sub-boxes with different density which compose it. Considering box B_1 of density ρ_1, and dimension X_1, box B_2 of density ρ_2, and dimension X_2 and box B of density ρ with:

$$\rho = \frac{X_1\rho_1 + X_2\rho_2}{X_1 + X_2}$$

We proceed as if the boxes B_1 and B_2 had a density of ρ: therefore B_1 becomes B_1' of length X_1' by compression and B_2 becomes B_2' of length X_2' by stretching (if $\rho_2 \geq \rho_1$), this does not change their moments. Thus it is the case of regrouping two boxes of identical density and this is achieved by using (19).

Extension to three dimensions

The extension of the definitions and of the equations (17), (18) and (19) to three dimensions is straightforward; for example, the equations (17) must be completed by:

$$M_z^D = \alpha\left[M_z + (1-\alpha)\, M_{xz}\right]$$

$$M_{xz}^D = \alpha^2 M_{xz}$$

$$M_{yz}^D = \alpha M_{yz}$$

$$M_{zz}^D = \alpha M_{zz}$$

Properties of the scheme

The Prather scheme possesses the following properties:

- Stability: the scheme (forward Eulerian) is stable in the limit $\alpha \leq 1$, which is rigorously equivalent to the CFL condition: $u\, dt/dx \leq 1$.
- Conservation: the formulation of the scheme in flux form assures the exact conservation of the total mass of each tracer.
- Accuracy: compared to finite difference schemes, for which the mixing ratio is only known at the centre of the gridbox, additional information is provided by the storage of the first and second order moments and the calculation of their evolution. Prather [1986] estimated that the conservation of second order moments conferred on his scheme an accuracy comparable to a fourth order finite difference scheme.

- Small diffusivity and local character: the scheme is well adapted to the representation of localised phenomena, because the advection is performed by exchange between adjacent boxes (therefore there are no "distance" effects with this scheme). It should be noted that there is no continuity condition imposed on the tracer distribution at the interfaces of the boxes.
- Upwind character: in contrast to centred finite difference schemes, advection by the Prather scheme of any feature only affects the grid boxes downstream of this feature.

CPU memory and time requirement

The major inconvenience of the Prather scheme, which explains why it is not more widely used, is the cost in CPU memory and also CPU time. For each advected tracer the scheme requires (for N dimensions):

- 1 array for the zero order moment
- N arrays for the first order moments
- $N(N+1)/2$ arrays for the second order moments

Thus, for a 3D model using the conservation of second order moments, there are 10 3D arrays required for each tracer (and 6 2D arrays for a 2D model).

If the available memory is too limited (for example in high resolution or with a large number of tracers), it should be noted that it is possible to "truncate" the scheme to the conservation of first order moments only, which limits the number of arrays per tracer to $N+1$, i.e. 4 in three dimensions. Of course, the accuracy of the scheme and the characteristic low diffusivity are affected by this operation; according to Prather [1986], the scheme truncated in this way, equivalent to the "slope-scheme" of Russell and Lerner [1981], has an accuracy comparable to that of a 2nd order finite difference scheme.

However, it should be noted that effective doubling of the resolution obtained by the conservation of 2nd order moments can be achieved at a lower cost in CPU memory (multiplication by a factor 2.5 in 3D) and in CPU time (depending on the problem) than by increasing the number of gridboxes.

13.4 Use in Spherical Geometry

The Prather scheme first requires a grid to be defined over the model domain and second the calculation of the mass fluxes at the interfaces of all of the gridboxes thus defined. Here we show how the scheme can be implemented in a transport model using spherical geometry, and then how the mass fluxes at the box interfaces can be calculated exactly when the wind field is read in as spectral coefficients.

The results obtained here can obviously be used in an "off-line" transport model forced by analyses or from output from a spectral model (i.e. TOMCAT), or alternatively for the "on-line" transport of tracers (e.g. water vapour) in a spectral GCM.

Definition of the grid

A rectangular grid in spherical geometry is defined by the latitudes of the north/south interfaces and the longitudes of the east/west interfaces, i.e. by a series of latitudes $\phi_k, k = 0, 1...K$ and longitudes $\lambda_i, i = 0, 1, ...I$. The gridboxes $B_{i,k}$ defined by $B_{i,k} = [\lambda_{i-1}, \lambda_i] \times [\phi_{k-1}, \phi_k]$ cover the whole globe so that:

$\phi_0 = -\frac{\pi}{2}$ (south pole)

$\phi_K = \frac{\pi}{2}$ (north pole)

$\lambda_0 = \lambda_I$ (periodicity in longitude).

Note that at no stage do we assume that the grid is regular. Any rectangular grid which satisfies the above conditions is suitable for use with the scheme within TOMCAT.

Grid associated with a Gaussian grid

A spectral truncation is associated with a grid in physical space, a Gaussian grid, where the physical and non-linear terms are calculated. When the CTM is coupled with winds from a spectral CGM it is natural to use this grid for the basis of the Prather scheme. This will avoid using two grids and the problems of interpolation between the two.

The definition of such a grid is made by use of the following:

- The boxes are centred in longitude on the Gaussian grid:

$$\lambda_i = 2\pi \frac{(i - .5)}{I}$$

  The grid is therefore regular in longitude, like the Gaussian grid.

- The latitudes ϕ_k are defined successively from $\phi_0 = -\pi/2$ by using the Gaussian weights ω_k and the formula:

$$\mu_k - \mu_{k-1} = \omega_k$$

  where $\mu_k = sin\,(\phi_k)$

Remembering that

$$\sum_{k=1,K} \omega_k = 2 \tag{29}$$

this defines a grid which satisfies the above conditions, for which the surface area of the grid is:

$$S_{i,k} = \frac{2\pi a^2}{I}\,\omega_k \tag{30}$$

In TOMCAT the latitudes at the interfaces of the grid are given by `DLAT2`. Values of these are contained in the file `grid_data` in `/home/martyn/SLIMCAT or the TOMCAT/SLIMCAT web page` for most common resolutions.

Calculation of the mass fluxes at the interfaces:

Suppose that the grid on the sphere has been defined with no further assumptions. Now suppose that we know the horizontal wind field that will force the model in terms of spectral coefficients of streamfunction ψ and velocity potential χ.

The relations between (χ, ψ) , (U, V) and (η, ζ) are the following:

$$\eta = \frac{1}{a^2\,(1-\mu^2)}\left[\frac{\partial U}{\partial \lambda} + \left(1-\mu^2\right)\frac{\partial V}{\partial \mu}\right] \tag{31}$$

$$\zeta = \frac{1}{a^2\,(1-\mu^2)}\left[\frac{\partial V}{\partial \lambda} - \left(1-\mu^2\right)\frac{\partial U}{\partial \mu}\right]$$

$$\eta = \nabla^2 \chi \tag{32}$$

$$\zeta = \nabla^2 \psi$$

$$U = \frac{\partial \chi}{\partial \lambda} + \left(1-\mu^2\right)\frac{\partial \psi}{\partial \mu} \tag{33}$$

$$V = \frac{\partial \psi}{\partial \lambda} - \left(1-\mu^2\right)\frac{\partial \chi}{\partial \mu}$$

In the following we will also use the zonal η^λ and meridional η^μ divergences defined by:

$$\eta = \eta^\lambda + \eta^\mu \tag{34}$$

$$\eta^{\lambda} = \frac{1}{a^2\left(1-\mu^2\right)}\frac{\partial U}{\partial \lambda} \tag{35}$$

$$\eta^{\mu} = \frac{1}{a^2}\frac{\partial V}{\partial \mu} \tag{36}$$

It is possible to calculate exactly the mass fluxes at the interfaces of a rectangular grid, on the condition that this grid is regular in longitude, but not necessarily in latitude (of which a Gaussian grid is a particular example).

1. Zonal mass flux:

   Let $F^{\lambda}_{i,k}$ be the mass flux across the interface between the boxes $B_{i,k}$ and $B_{i+1,k}$.

   By definition:

   $$F^{\lambda}_{i,k} = \int_{\phi_{k-1}}^{\phi_k} \rho u a d\phi \tag{37}$$

   which can also be written:

   $$F^{\lambda}_{i,k} = \int_{\mu_{k-1}}^{\mu_k} \rho \frac{1}{(1-\mu^2)} U d\mu$$

2. Meridional mass flux:

   Let $F^{\phi}_{i,k}$ be the mass flux across the interface between the boxes $B_{i,k}$ and $B_{i,k+1}$.

   By definition:

   $$F^{\phi}_{i,k} = \int_{\lambda_{i-1}}^{\lambda_i} \rho v a cos\phi d\lambda \tag{38}$$

   which can be written:

   $$F^{\phi}_{i,k} = \int_{\lambda_{i-1}}^{\lambda_i} \rho V d\lambda$$

If we consider the box $B_{i,k}$, our problem is therefore to calculate: $F^{\lambda}_{i-1,k}$, $F^{\lambda}_{i,k}$, $F^{\mu}_{i,k-1}$ and $F^{\mu}_{i,k-1}$.

a) calculation of the meridional flux:

The first step consists of calculating the Fourier coefficients of V at the north-south interfaces of the grid by using the values of P_{nm} tabulated at the latitudes of the interfaces.

For example:

$$V_m\left(\mu^-\right) = \sum_n V_n^m Y_n^m \tag{39}$$

Performing a direct Fourier transform on these coefficients will give us the values of V at the points regularly spaced in longitude. Instead of this, note that we can use the Fourier decomposition of V:

$$V(\lambda,\mu) = \sum_m V_m(\mu)\, e^{im\lambda}$$

to write:

$$\int_{\lambda_{i-1}}^{\lambda_i} V d\lambda = \sum_m V_m(\mu) \int_{\lambda^{i-1}}^{\lambda_i} e^{im\lambda} d\lambda \tag{40}$$

If we note $\lambda_i^G = \frac{1}{2}\left(\lambda_{i-1} + \lambda_i\right)$ the longitudes of the Gaussian grid, and $\Delta\lambda$ the longitude increment of this grid, we can easily transform the integrals to:

$$\int_{\lambda_{i-1}}^{\lambda_i} e^{im\lambda} d\lambda = \frac{1}{im} e^{im\lambda}\left(e^{\frac{im\Delta\lambda}{2}} - e^{\frac{-im\Delta\lambda}{2}}\right) \tag{41}$$

$$\int_{\lambda_{i-1}}^{\lambda_i} e^{im\lambda} d\lambda = \frac{2}{m} sin\left(\frac{m\Delta\lambda}{2}\right) e^{im\lambda}$$

and finally we obtain, for the meridional mass flux F, a formula analogous to that giving V, with a multiplication of the Fourier coefficients V_m by precalculated factors:

$$F_m = V_m \frac{2}{m} sin\left(\frac{m\Delta\lambda}{2}\right)$$

After calculating V_m as indicated above, we therefore multiply them by a factor precalculated before performing the FFT which leads exactly to the integrated value of the meridional mass flux at the interfaces.

$$F^{\phi} = \sum_m F_m e^{im\lambda}$$

This method requires the prior tabulation of the Legendre functions $P_{nm}(\mu)$ and their derivatives $(1-\mu^2)dP_{nm}/d\mu$ at the latitudes of the specified box interfaces (in the file `LEGIyy` - see section xx.4).

b) calculation of the zonal flux:

We proceed differently for the calculation of the zonal mass flux. If we consider the same grid as before, we will first calculate the divergence of the zonal mass flux, obtained from the total divergence of the mass flux minus the meridional divergence.

The total divergence of the mass flux is calculated by using a similar approach to that for the calculation of the meridional mass flux. The meridional divergence is obtained immediately by the difference between the meridional fluxes at the north and south interfaces of the box.

Finally, we obtain the divergence of the zonal mass flux, which allows us to calculate in turn the zonal flux itself at the east/west interfaces of the gridboxes, in terms of a constant from the condition of periodicity in longitude. This constant is determined from the zonal mean of the zonal mass flux.

This method requires the prior tabulation of the values of the Legendre functions averaged over the boxes in the employed grid (also in the file `LEGIyy`).

Note that it would be possible to calculate directly the zonal flux without using the divergence, by using the formula:

$$\frac{U}{(1-\mu^2)} = \frac{1}{(1-\mu^2)} \frac{\partial\chi}{\partial\lambda} + \frac{\partial\psi}{\partial\mu} \tag{42}$$

but this requires the additional tabulation of the derivatives of P_{nm}

This allows the calculation in terms of a constant (from the periodic condition over the longitudes). This constant is determined from:

$$\sum_i F^{\lambda}_{i,k} = \sum_i F^{\lambda}_{i,k} \tag{43}$$

i.e. the calculation of the zonal mean of the zonal wind from the values at the centres of the boxes or from the values at the interfaces leads to the same result.

Thus, the total mass flux $F_{i,k}$ entering a box $B_{i,k}$ is given by:

$$F_{i,k} = F^{\lambda}_{i-1,k} - F^{\lambda}_{i,k} + F^{\phi}_{i,k-1} - F^{\phi}_{i,k}$$

and we can verify that

$$F_{i,k} = -S_{i,k}\,\bar{\eta_{i,k}} \tag{44}$$

High latitudes

The stability of the Prather scheme is conditional on the constraint: $\alpha \leq 1$, rigorously equivalent to the CFL condition: $u\Delta/\Delta x \leq 1$ (for advection along Ox). While this constraint does not pose any particular problems and allows the use of reasonable timesteps for the advection along latitudes or in the vertical, it is not the same for the advection along longitude, because of the convergence of the meridians towards the poles and the reduction of Δx which follows ($\Delta x = acos(\phi)\Delta\lambda$). To avoid very small timesteps, the model uses "extended polar zones", as named by *Prather et al.* [1987] at high latitudes. This correction consists of grouping several adjacent boxes situated on the same latitude circle and advecting this block along the longitude in the same way as a single box of larger size. The number of boxes grouped together depends on the latitude, on the size of the boxes $\Delta\lambda$, on the timestep Δt, as well as on the maximum wind that will be encountered. The number of boxes grouped together at each latitude is `NUM` which is set up in the routine `CALNUM`. Before the advection step in the x direction the boxes are regrouped by using (17), then, when the zonal advection is complete, the original boxes are reconstructed using the reciprocal formulae (19).

Problem of the flux across the pole

As shown above the meridional mass flux F_ϕ across an element of surface situated at latitude ϕ is given by:

$$F^{\phi}_{i,k} = \int_{\lambda_{i-1}}^{\lambda_i} \rho va.cos(\phi).d\lambda \tag{45}$$

In particular, the mass flux across the pole is zero even if the cross-polar wind is not zero. The quantity of mass passing directly from one polar gridbox to the gridbox diametrically opposite the pole is therefore not always zero. The mass flux from one box to the other occurs by crossing one by one the neighbouring boxes, therefore it is a case of a zonal mass flux and not a meridional one.

Unfortunately, the zonal flow to a neighbouring box is poorly represented near the pole in the Prather scheme as discussed above. In effect, one of the characteristics of the scheme is that the mass flux calculated at the interfaces of the boxes are spread over these interfaces as of they were due to a uniform velocity (this does not mean that the calculation of the mass flux does not take account of the wind shear on the interface). Thus, in planar geometry, the volumes transported from one box to another are rectangular slabs, the velocity being assumed uniform over the sides of the boxes; in our case of interest (spherical geometry) the volumes transported along longitude are in reality angular sectors, which is equivalent to considering the angular velocity $\alpha = d\lambda\ dt = u/cos(\phi)$ constant over the interface. In particular, for the boxes neighbouring the pole, the motion is that of a rotation about the pole, which returns to taking a zero velocity at that point. In summary, the scheme implicitly considers in part that the mass flux across the pole is zero, which is true, but equally that the velocity at the pole is zero, which is not true in general.

In fact, the operation of uniformly spreading the mass flux over the whole length of the interface is equivalent to substituting the real wind field with a discretised one which has the following properties:

- The mass flux across the interfaces of the grid are the same with the discretised wind field as with the real wind field. In particular, if the real wind field is non divergent, the same is true of the discretised wind field.
- The meridional velocity v is uniform over the north/south interfaces.
- The zonal velocity $u/cos(\phi)$ is uniform over the east/west interfaces.

In addition, it follows that the discretised wind field has discontinuities at the interfaces of the gridboxes: there is a discontinuity in the meridional component at the east/west interfaces and a discontinuity in the zonal component at the north/south interfaces.

The problem of the treatment of the pole (a singularity) in a gridpoint model is a classical problem. A test to verify the performance of a numerical advection scheme in the case of cross polar flow is to consider a

solid rotation about and axis inclined at 90 degrees to the polar axis. This test has been used, for example, by Williamson and Rasch [1989] for a semi-lagrangian scheme, taking an initial tracer distribution of a localised structure at the equator of the solid rotation.

The equation of the motion is the following:

$$u = Usin\lambda cos\phi$$

$$v = -Usin\lambda$$

As the motion is a solid rotation, the initial structure moves on the sphere without being deformed and returns to its initial position after travelling around the globe and crossing the two poles.

A method sometimes employed to solve this deficiency is to cover the pole by a "polar box", a circular zone centred on the pole inside which the mixing ratio of tracers is taken to be uniform. Therefore the treatment of the singular point is avoided. This method has been used by *Rood et al.* [1991] with the Van Leer scheme and seems capable of giving good results. Therefore it was tested with the Prather scheme by taking the polar box to be the ensemble of boxes which touch the pole. However, this approach gave transport across the pole which was too rapid. Therefore, in TOMCAT the mass flux across the pole is obtained by calculating the wind vector at the pole and determining the mass of each triangular box which would be transported to the diametrically opposite box in each timestep. This then gives the mass flux across the pole.

13.5 Correction of Negative Values

As advection schemes are not perfect it is necessary to couple them with a certain number of fixes to correct their intrinsic deficiencies. A well-known problem with classical transport schemes (spectral and finite difference) is their non-positiveness. For minor constituents (e.g. water vapour, ozone) it is obvious that negative values are purely a numerical artefact and that these negative values can be a problem in chemical or physical calculations.

Contrary to what is sometimes assumed, the formulation of the Prather scheme does not ensure the positiveness of a tracer without an additional correction. This can be seen in the example in figure 3 of Prather [1986]. In a one dimensional example, if the tracer at t_0 is localised in a single box B_i of the grid (with the 1st and 2nd order moments set to zero), and the wind u_i is directed towards the right.

$$M_0^i = 1 \tag{46}$$

$$M_x^i = 0$$

$$M_{xx}^i = 0$$

$$M_0^{i+1} = 0 \tag{47}$$

$$M_x^{i+1} = 0$$

$$M_{xx}^{i+1} = 0$$

After a timestep dt, the structure is globally displaced towards the right, and only the distributions of the boxes B_i and B_{i+1} are modified; the distribution at $t_0 + dt$ in the box B_{i+1} calculated using (17),(18) and (19) is the following:

$$M_0^i = (1 - \alpha) \tag{48}$$

$$M_x^i = 3\alpha\,(1 - \alpha)$$

$$M_{xx}^i = 5\alpha\,(2\alpha - 1)\,(1 - \alpha)$$

$$M_0^{i+1} = \alpha \tag{49}$$

$$M_x^{i+1} = -3\alpha\,(1 - \alpha)$$

$$M_{xx}^{i+1} = -5\alpha\,(2\alpha - 1)\,(1 - \alpha)$$

At the following timestep, for certain values of the wind α, the quantity of tracer mass which will be transported into the box B_{i+2} will be negative, which, bearing in mind the absence of tracer in this box at $t_0 + dt$ will create a negative values at the instant $t_0 + 2dt$. As can be seen, the appearance of negative values is due to the fact that during the reconstruction of the boxes at $t_0 + dt$ using the formulae (19), a distribution is created which is globally positive ($M_0 \geq 0$ even in the box B_{i+1}), but not the local positiveness, allowing the advection of negative quantities of mass. Therefore, this problem can be solved by correcting the first and second order moments before this advection to give a distribution which is positive everywhere: the correction proposed by Prather [1986] consists of:

$$M_x^* = min\left[\frac{3}{2}M_0, max\left(-\frac{3}{2}M_0, M_x\right)\right] \tag{50}$$

$$M_{xx}^* = min\left[2M_0 - \frac{1}{3}|M_x^*|, max\left(|M_x^*| - M_0, M_{xx}\right)\right] \tag{51}$$

Note that this correction slightly increases the diffusivity of the initial scheme, but this effect is very small as the creation of negative values as described above is comparatively rare and the correction consists of a redistribution of tracer mass within a gridbox. This correction is termed "flux limiting" as it consists of avoiding negative mass fluxes. However, the use of such a flux limiter will destroy tracer correlations. Generally, it is best to avoid the use of the limiter if negative tracer values can be tolerated.

14 Appendix 4. Chemically Updating Tracer Moments

When using the *Prather* [1986] advection scheme in the basic TOMCAT/SLIMCAT model, any change in tracer mass due to chemistry only acts upon the zero order tracer moment (`S0`). Strictly, the first and second order moments should also be modified. This would increase the chemical resolution of the model and could reduce the rate of chemical reaction between two tracers with opposite gradients within a gridbox, for example. However, chemically updating the 1st and 2nd order moments would be prohibitively expensive in a full chemistry simulation.

Consider the integrals over a gridbox (in two dimensions) of the products of any 3 basis functions of the moments (equation (11) above). If one of the functions is the zero order moment (K_o) we get:

$$\int_0^X \int_0^Y K_0 K_0^2\,(x,y)\,dxdy = XY = S \tag{52}$$

$$\int_0^X \int_0^Y K_0 K_x^2\,(x,y)\,dxdy = S/3 \tag{53}$$

$$\int_0^X \int_0^Y K_0 K_y^2\,(x,y)\,dxdy = S/3 \tag{54}$$

$$\int_0^X \int_0^Y K_0 K_{xx}^2\,(x,y)\,dxdy = S/5 \tag{55}$$

$$\int_0^X \int_0^Y K_0 K_{yy}^2\,(x,y)\,dxdy = S/5 \tag{56}$$

$$\int_0^X \int_0^Y K_0 K_{xy}^2\,(x,y)\,dxdy = S/9 \tag{57}$$

Otherwise, the integrals give

$$\int_0^X K_x^3(x,y)\,dx = 0 \tag{58}$$

$$\int_0^X K_x(x,y)\,K_{xx}^2(x,y)\,dx = 0 \tag{59}$$

$$\int_0^X K_x^2(x,y)\,K_{xx}(x,y)\,dx = 2S/15 \tag{60}$$

$$\int_0^X K_{xx}^3(x,y)\,dx = 2S/35 \tag{61}$$

$$\int_0^Y \int_0^X K_x(x,y)\,K_y(x,y)\,K_{xy}(x,y)\,dxdy = S/9 \tag{62}$$

$$\int_0^Y \int_0^X K_{xy}^2(x,y)\,K_{xx}(x,y)\,dxdy = 2S/45 \tag{63}$$

Similar results are obtained for K_y and K_{yy}, or K_z and K_{zz}. All of the other products involving the 'cross terms' (mixtures of x,y, and z) are zero (see equation (12) above).

Consider two chemical tracers whose distributions in two dimensions are given by:

$$r_A(x,y) = \frac{1}{M}\sum_\alpha A_\alpha K_\alpha(x,y) \tag{64}$$

and

$$r_B(x,y) = \frac{1}{M}\sum_\alpha B_\alpha K_\alpha(x,y) \tag{65}$$

The rate of the chemical reaction between the two species depends on the product of the concentrations:

$$rate = k[A][B]$$

$$rate = k^{'}(A_0K_0+A_xK_x+A_{xx}K_{xx}+A_yK_y+A_{yy}K_{yy}+A_{xy}K_{xy})(B_0K_0+B_xK_x+B_{xx}K_{xx}+B_yK_y+B_{yy}K_{yy}+B_{xy}K_{xy})$$

The effect on the zero order moment is proportional to:

$$\int_0^X \int_0^Y ABK_0\,dxdy = A_0B_0 + \frac{1}{3}(A_xB_x + A_yB_y) + \frac{1}{5}(A_{xx}B_{xx} + A_{yy}B_{yy}) + \frac{1}{9}A_{xy}B_{xy} \tag{66}$$

So, in addition to the product of the average concentration of A with the average concentration of B (first term on RHS), the zero order moments of A and B are modified by products of the higher order moments.

The effect on the first order moment in the x direction is proportional to:

$$\int_0^X \int_0^Y ABK_x\,dxdy = \frac{1}{3}(A_xB_0 + A_0B_x) + \frac{2}{15}(A_xB_{xx} + A_{xx}B_x) + \frac{1}{9}(A_yB_{xy} + A_{xy}B_y) \tag{67}$$

and similarly for the y direction.

The effect on the second order moment in the x direction is proportional to:

$$\int_0^X \int_0^Y ABK_{xx}\,dxdy = \frac{1}{5}(A_{xx}B_0 + A_0B_{xx}) + \frac{2}{15}(A_xB_x) + \frac{2}{35}(A_{xx}B_{xx}) + \frac{2}{45}(A_{xy}B_{xy}) \tag{68}$$

The effect on the second order moment in the xy direction is proportional to:

$$\int_0^X \int_0^Y ABK_{xy} dx dy = \frac{1}{9}\left(A_0 B_{xy} + A_{xy} B_0\right) + \frac{1}{9}\left(A_y B_x + A_x B_y\right) + \frac{2}{45}\left(A_{xy} B_{xx} + A_{xx} B_{xy} + A_{yy} B_{xy} + A_{xy} B_{yy}\right) \tag{69}$$

Therefore, to chemically update the 0, 1st and 2nd order moments of a chemical tracer in three-dimensions would require integrating the chemistry 10 times. As chemical integration is generally by far the most costly part of a CTM, this is prohibitively expensive. However, it may be practical to use the higher order moments to update just the zero order moments (equation (57)).

15 Appendix 5. MPI version of TOMCAT/SLIMCAT

As described the Section 2.5, the current default version of the model is parallelised using OpenMP. This will run on many platforms although it is possible that in the future hardware issues may limit high-end HPC machines to distributed memmory parallelisation. Therefore, I have created a copy of the standard library (UNICATMPP0.xx) which is identical to the standard library as far as possible but with MPI parallelisation. This MPI version is based on older versions of parallel TOMCAT and SLIMCAT developed by Cate Bridgeman. This appendix describes things specific to this MPI library.

Not all of the UNICAT options can be used under MPI. The most likely use of the MPI library is to perform otherwise time consuming runs. Therefore, the MPI library is largely restricted to the default options (e.g. Prather second order moments advection) and the most expensive configurations of the model (e.g. full chemistry).

15.1 Domain Decomposition

LAT \ LON	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	0	0	1	1	2	2	3	3	4	4	5	5	6	6	7	7
2	0	0	1	1	2	2	3	3	4	4	5	5	6	6	7	7
3	8	8	9	9	10	10	11	11	12	12	13	13	14	14	15	15
4	8	8	9	9	10	10	11	11	12	12	13	13	14	14	15	15
5	16	16	17	17	18	18	19	19	20	20	21	21	22	22	23	23
6	16	16	17	17	18	18	19	19	20	20	21	21	22	22	23	23
7	24	24	25	25	26	26	27	27	28	28	29	29	30	30	31	31
8	24	24	25	25	26	26	27	27	28	28	29	29	30	30	31	31

Figure 4: Computational grid for MPI model showing the domain owned by processor 12 and a possible halo.

The MPI library has been parallelised in the "Single Program Multiple Data" (SPMD) style which means that the same program is run by each processor, but using its own data. This is achieved in UNICATMPP by splitting the horizontal grid into **patches** so that each processor owns a unique subset of grid points, and performs computations just on its own patch. Sometimes a processor will need to know about points owned by neighbouring processors. This is most important for the advection step when the processor needs to work out how much of a particular tracer has moved into its patch from outside. This is done by creating a **halo** of grid points around each patch which the processor is aware of but does not perform any computations on. This is illustrated in figure 4 for processor 12 of 32 on a grid of 8 `LAT` x 16 `LON`. In order to keep the data associated with these halo points up to date, the processors need to communicate with their neighbours before each advection time step. The size of halo required depends on the length of the time step and the resolution of the grid since it must contain all points from which tracers may enter the domain during the time step. The time taken to perform the communication depends both on the size of the message to be communicated and on a startup time during which the processors set up a communication channel. Thus communication time increases as the time step increases and, more importantly, as the number of processors increases. The latter occurs for two reasons. Firstly more processors means more halos to be filled and secondly communication may be required not just between adjacent processors but also between more remote neighbours. This can result in large overheads and a consequent decrease in the efficiency of the program. Communication is achieved in UNICATMPP via the message passing interface (MPI).

In principle it is possible to parallelise the code to allow for irregular (non-rectangular) patches of different sizes. Cate Bridgeman's orginal code allowed for this in certain (but not all!) routines. However, for ease and to keep the UNICAT MPI version as similar as possible to the standard version, the model will now only deal with rectangular patches of equal size. In practice this means that the model grid (LON, LAT) should be values which are convenient multiples of the factors of the number of processors requested. Because standard grids tend to have 32, 64, 128 etc longitudes / latitudes etc, and CPUs often come in multiples of 16, in effect this is not a restriction at all.

15.2 Defining the Grid

The model resolution is set in the deck PARADI. Additional parameters (not used in the Scalar/OpenMP model) define the decomposition of the grid.

Parameter	Meaning
NLATMX	Number of latitudes in a patch (excluding halo)
NLONMX	number of longitudes in a patch (excluding halo)
NHALMX	Max extent of halo in gridpoints
NLATPP	Number of processes per latitude
NPROCMAX	Maximum number of processes
NPROCI	Number of processes along longitude row
NPROCK	Number of processes along latitude row

NLONMX and NLATMX are the maximum number of longitude and latitude points owned by any processor. NPROCI defines the number of processors along a horizontal row and so NLONMX=LON/NPROCI. NHALMX defines the maximum number of halo points that are required at a single latitude.

```
 PARAMETER( NLONMX = 32,                                    PARADI.68
$           NLATMX = 4,                                     PARADI.69
$           NHALMX = 300,                                   PARADI.70
$           NPROCI = 4                                      PARADI.71
$           NPROCK = 2 )                                    PARADI.72
```

NPROCMAX is the total number of processors to be used.

```
 PARAMETER( NPROCMAX = 64,                                  PARADI.76
$           NSLSTMX  = 3000,                                PARADI.77
$           NRLSTMX  = 3000,                                PARADI.78
$           NFLDSL   = 220,                                 PARADI.79
```

NSLSTMX specifies the maximum number of grid points any processor will need to send information about to any other processor in preparation for the advection step, and NRLSTMX is the equivalent buffer size for receiving data. The total number of model fields to be exchanged prior to advection is given by NFLDSL which is set to MAX(3,NIV*(1+10*NTRA)) for Prather SOM advection.

For most choices of processor number, there are several possible grid decompositions, and the possibilities can be calculated using the program **calc_slcons.f** which is in the **/home/martyn/UNICAT** directory. The parameters in the program need to be sent to match the actual TOMCAT/SLIMCAT job for LON, LAT, NIV, NTRA, NDYN, DT0 and UMAX. The program then ouputs the possible domain decompositions, e.g.:

Partial output from calc_slcons.f
Possible choices for SLCONS are :-
Number of processors: 64

nlonmx	= 8
nlatmx	= 4
nhalmx	= 8
nproci	= 8
nprock	= 8
nprocmax	= 64
nslstmx	= 48
nrlstmx	= 48
nfldsl	= 8401
nlist	= 2

nlonmx	= 16
nlatmx	= 2
nhalmx	= 8
nproci	= 4
nprock	= 16
nprocmax	= 64
nslstmx	= 56
nrlstmx	= 56
nfldsl	= 8401
nlist	= 2

Number of processors: 32
...

The model generally runs quicker if the patches are as square as possible as this minimises communication between processors, so choose a decomposition where `NLONMX` and `NLATMX` are similar lengths.

15.3 Compiling and Running

(As for normal model - get an example job deck)

15.4 MPI Performance

The execution time T of a parallel program (defined as the time elapsed between the first processor beginning and the last processor completing the task) depends on a number of variables. The principal ones are the size of the problem, N (grid resolution and number of layers and tracers in this case) and the number of processors P. Execution time is often not the most effective way of assessing performance. Since T varies with problem size, even on a single processor, it is difficult to compare performance at different N. Efficiency, E, removes this size dependence and is defined as

$$E = T_1/PT_P,$$

Table 1: CPU time, speedup and efficiency for a single tracer at T42.

NPE	time (s)	speedup	efficiency
1 (fuji)	9483		
1 (T3E)	7530	1.00	1.00
2	3700	2.04	1.02
4	1900	3.96	0.99
8	943	7.99	1.00
16	504	14.98	0.94
32	296	25.44	0.80
64	213	34.35	0.54
128	192	39.23	0.31

Table 2: CPU time, speedup and efficiency for MPI job with a single tracer at T42.

NPE	time (s)					speedup	efficiency
	initialisation	pbl	transport	chemistry	total	(compared to 4PE)	
fuji					4704		
4	65	270	1395	3595	5325		
8	49	138	1242	1853	3281	6.49	0.81
16	23	77	741	875	1716	12.41	0.78
32	7	65	491	393	955	22.30	0.70
64	9	43	425	231	708	30.09	0.47

where T_P is the execution time on P processors. Speedup S is the factor by which execution time is reduced on P processors,

$$S = PE_P.$$

Table 1 shows CPU time in seconds, speedup and efficiency for a 30 day run at T42 and 31 levels using a single methane tracer, simple chemistry and Louis vertical diffusion scheme. For this run speedup increases up to 64 processors, but by this point communication costs outweigh the advantage of each processor dealing with fewer grid points and speedup is not substantially greater than that achieved using half as many processors. Efficiency is nearly 100% for up to 16 processors and then begins to fall as input/output overheads and communication costs become more important.

Table 2 shows CPU time in seconds for a 2-day run at T21 and 31 levels with `NTRA=27` and `NVTOT=49` using the PBL scheme and ASAD chemistry scheme. 32 processors is a sensible choice for a run of this sort. In moving from 32 to 64 processors the speedup increases by 35% but efficiency drops to below 50%. It is apparent from the above times that the bottleneck occurs in the transport step. There are two reasons for this. Firstly it is the transport step that requires most of the communication between processors. Secondly, and more importantly, quantities from the ECMWF forcing files undergo Fourier transforms before they are used in TOMCAT. These transforms must be performed on an entire latitude and so a T21 run (with 32 latitudes) can only share the work between 32 processors. Any additional processors will be idle while waiting for a transform to be completed and this results in loss of efficiency.

16 Appendix 6. Writing Fortran for TOMCAT/SLIMCAT

TOMCAT/SLIMCAT are written in Fortran and in a certain style. By following a few rules and conventions, any new code should be easier to understand, tidier, more efficient and with a reduced risk of containing errors.

This document explains these rules and conventions. Section 1 describes good practice for all Fortran programs. Section 2 gives information specific to TOMCAT/SLIMCAT.

16.1 General Rules in Fortran

16.1.1 When Writing Code

When writing new Fortran code you should always:

- Declare all variables explicitly. Use `IMPLICIT NONE` at the start of each subroutine. However, it is still good practice to use variables beginning with `I-N` for `INTEGER`s and other letters for `REAL`s.
- Declare all constants as `PARAMETER`s.
- Avoid using variables of 1 character (except for loop indices).
- Simplify mathematical expressions as much as possible before coding them up.
- Avoid very short subroutines.
- Reduce the use of intrinsic functions such as `EXP`, power (`X**Y`), etc which can be expensive, by tidying up expressions and storing and reusing part-expressions.
- Read input files only once (and store numbers if necessary).
- Avoid large levels of nested loops.

16.1.2 When Testing Code

- Get rid of all compiler warning messages. Compile the code using different compilers to help get rid of non-standard features and make it portable.
- Use the compiler option to set all variables as 'undefined' (e.g. `frt -Hu`).
- Test the array bounds and consistency of subroutine calls (e.g. `frt -Haes`). (However this usually has a large overhead so turn this off for real experiments).

The most important point is to write correct code. Once this code is running there is software (e.g. `ssrun` on Green/Wren/Fermat) which shows where the model is spending its time. Then the code can be speeded-up and optimised (taking care not to affect the results!).

16.2 Styles/Conventions in TOMCAT/SLIMCAT:

Most of the code in TOMCAT/SLIMCAT follows a certain style. Given the size of the existing code, compared to anything you may add, it will be better if your new code fits in with this style.

(Exceptions to the generic style occur if whole modules of code are coupled with the model. For example, the CCM radiation scheme (and ASAD) is written in a different style (with lower case variables) and so where these variables are referenced within TOMCAT lower case is also used).

I would say the conventions that define the TOMCAT/SLIMCAT style are:

- The code is written in upper case. The comments are written in normal text (i.e. mixture of upper/lower case).
- Code within loops and in `IF/ENDIF` blocks (etc.) should be indented 2 spaces.
- Three-dimensional arrays are declared as (`LON, LAT, NIV`). The standard loop indices for these 3 dimensions are I, K, and L respectively. JV is used for tracers.
- Only use `COMMON` blocks in conjunction with nupdate common decks or included files (so that code is only in 1 place).
- Don't insert nupdate common decks in a SUBROUTINE unnecessarily.

16.3 Example Subroutine

The following subroutine illustrates this style.

```
      SUBROUTINE EXAMPLE(AIN,BIN,COUT,NDIM)
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C     Calculate <....>
C
C     Method:
C     -------
C     <description>
C
C     Input:
C     ------
C     AIN         <description, units>
C     BIN         <description, units>
C
C     Output:
C     -------
C     COUT        <description, units>
C
C     References
C     ----------
C     <details of any papers/reports etc>
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
```

```
      IMPLICIT NONE
C
*CALL,PARADI
*CALL,MOMENT
*CALL,FOR3D
C
C     Arguments
      INTEGER NDIM
      REAL AIN(NDIM),BIN(NDIM),COUT(NDIM)
C
C     Local arrays
      REAL SUM
      INTEGER I,K,L,N
C
C
C     Calculate mass-weighted mean global T (K)
      SUM=0.0
      DO L=1,NIV
      DO K=1,LAT
      DO I=1,LON
        SUT=SUT + T3D(I,K,L)*SM(I,K,L)
        SUM=SUM + SM(I,K,L)
      ENDDO
      ENDDO
      ENDDO
      SUT=SUT/(SUM*FLOAT(LON*LAT*NIV))
C
C     Scale...
      DO N=1,NDIM
        COUT(N)=AIN(N) + BIN(N)*SUM
      ENDDO
C
      RETURN
      END
```

16.4 Example Interface

A model generally consists of different modules (i.e. collection of subroutines) to calculate different processes in the atmosphere (e.g. chemistry, radiation, microphysics). Although these modules need to be coupled, the code in each module should be largely independent. Indeed, it is an advantage to have a module mostly self-contained as it will be more portable and can be more easily coupled with other models.
In TOMCAT/SLIMCAT many parts of the model are completely entwined, for example the code to read the analyses, calculate mass fluxes output the results. This is really the core of the model and makes large use of nupdate common decks to pass variables around. However, when adding new modules it is better if they interface with TOMCAT/SLIMCAT in just 1 routine.
The module can still use nupdate common blocks to store/pass its own variables, but it is good to avoid using TOMCAT/SLIMCAT ones. Variables can be passed as arguments. If a few constants are needed, rather than calling the TOMCAT/SLIMCAT deck `CSTUNI` the constant can be declared as a local `PARAMETER` (and given the same name/value).

```
      SUBROUTINE INTERF
```

```
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
C
C     Example interface routine
C
CCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCCC
      IMPLICIT NONE
C
*CALL,PARADI
*CALL,MOMENT
*CALL,FOR3D
C
      INTEGER NDIM,NAER
      PARAMETER(NDIM=LON*NIV, NAER=2)
C
C     Aerosol fields
      REAL T1D(NDIM),P1D(NDIM),SO1D(NDIM),SO1D(NDIM,NAER)
C
C     Read in fields on LAT,LON (and NIV) grid here.
C
C     Loop over latitude (can be parallelised)
      DO 100 K=1,LAT
C
C       Copy TOMCAT 3D arrays for e.g. tracers, T, p to
C       arrays for aerosol calculation
        DO L=1,NIV
        DO I=1,LON
          II=(L-1)*NIV + LON
          T1D (II)=T3D(I,K,L)
          P1D (II)=PL (I,K,L)
          SO1D(II,1)=SO(I,K,L,10)
          SO1D(II,2)=SO(I,K,L,11)
        ENDDO
        ENDDO
C
C       Call aerosol scheme
        CALL AEROSO(T1D,P1D,SO1D,NDIM,NAER)
C
C       Copy aerosol results to TOMCAT tracers
        DO L=1,NIV
        DO I=1,LON
          II=(L-1)*NIV + LON
          SO(I,K,L,10)=SO1D(II,1)
          SO(I,K,L,11)=SO1D(II,2)
        ENDDO
        ENDDO
C
      End of (parallel) latitude loop
  100 CONTINUE
C
      RETURN
      END
```

17 Appendix 7. Flowtrace and other performance tests

Below is a flowtrace from a 1 day multilevel SLIMCAT run using the CCM radiation scheme with `NIV=13, LON=64, LAT=32` with (24-hourly) UKMO winds and `LVERT=.TRUE.`.

This can be compared with an equivalent run using MIDRAD:

17.1 TOMCAT - Cray YMP8

Below is a flowtrace on a Cray YMP8 from a 1 day run with `NIV=19, LON=128, LAT=64` with (6-hourly) ECWMF winds and 2 tracers. The job included convection and vertical diffusion.

```
+ flowview -Luc
                        Flowtrace Statistics Report
              Showing Routines Sorted by CPU Time (Descending)
                      (CPU Times are Shown in Seconds)

Routine Name     Tot Time  # Calls Avg Time Percentage  Accum%
---------------- -------- -------- -------- ----------  -------
ADVX2            4.03E+01       96 4.20E-01      22.88    22.88  *****
ADVY2            3.40E+01       96 3.54E-01      19.28    42.15  ****
CONSOM           2.73E+01       48 5.69E-01      15.49    57.64  ***
ADVZ2            1.59E+01       48 3.32E-01       9.02    66.67  **
CALSUB           1.31E+01        4 3.28E+00       7.43    74.10  *
CONVMA           9.88E+00    32768 3.01E-04       5.60    79.70  *
MUHERM           5.76E+00   131072 4.40E-05       3.27    87.58
CHIMIE           4.74E+00       48 9.87E-02       2.69    90.26
CALFLU           3.80E+00        5 7.59E-01       2.15    92.42
INITER           2.39E+00       48 4.98E-02       1.36    93.77
CLOUD            1.94E+00    32768 5.91E-05       1.10    94.87
FINCYCL          1.80E+00        4 4.51E-01       1.02    95.89
QSAT             1.38E+00   292132 4.71E-06       0.78    96.67
DQSATDT          8.62E-01   168175 5.13E-06       0.49    97.16
LOUIS            6.44E-01    32768 1.96E-05       0.36    97.53
PELF             5.85E-01        5 1.17E-01       0.33    98.22
SUBSCAL          5.05E-01    32768 1.54E-05       0.29    98.51
REEMDT           4.55E-01        1 4.55E-01       0.26    98.76
PEFL             4.51E-01      160 2.82E-03       0.26    99.02
PEFP             3.27E-01      320 1.02E-03       0.19    99.20
TOMCAT           2.63E-01        1 2.63E-01       0.15    99.35
INIEXP           2.58E-01        1 2.58E-01       0.15    99.50
REEZNOT          2.49E-01        1 2.49E-01       0.14    99.64
PEPF             2.46E-01      320 7.70E-04       0.14    99.78
ADVEC            1.67E-01       48 3.49E-03       0.09    99.88
INICYCL          7.08E-02        4 1.77E-02       0.04    99.92
CEP              3.25E-02      320 1.01E-04       0.02    99.93
CEF1             3.04E-02      320 9.51E-05       0.02    99.95
CONVEC           2.33E-02       48 4.85E-04       0.01    99.96
ALRET            2.27E-02        5 4.54E-03       0.01    99.98
```

```
CEF2             1.26E-02      320 3.92E-05       0.01    99.98
CORPOLE          8.40E-03        5 1.68E-03       0.00    99.99
CHTRON           7.03E-03      385 1.83E-05       0.00    99.99
FINITER          6.79E-03       48 1.41E-04       0.00   100.00
INICSF           2.64E-03        5 5.27E-04       0.00   100.00
WRCHK            2.12E-03        1 2.12E-03       0.00   100.00
FINEXP           8.28E-05        1 8.28E-05       0.00   100.00
INICSTE          3.46E-06        1 3.46E-06       0.00   100.00
INCHK            2.26E-06        1 2.26E-06       0.00   100.00
================================================================================
Totals           1.76E+02   725361
Jun 12 19:31 mpc        77.95 Mflops    191.93s a.out
```

The most expensive routines are the second order moments advection scheme and `CONSOM` which applies the convection/diffusion matrix to the tracers. However, the code is written efficiently for the Cray.

17.2 SLIMCAT - Cray YMP8

Below is a flowtrace from a 10 day single level run with `NIV=1, LON=128, LAT=64` with (24-hourly) UKMO winds and `LVERT=.FALSE.`.

```
+ flowview -Luc

                    Flowtrace Statistics Report
             Showing Routines Sorted by CPU Time (Descending)
                    (CPU Times are Shown in Seconds)

Routine Name     Tot Time  # Calls Avg Time Percentage  Accum%
---------------- -------- -------- -------- ----------  -------
ADVX2            1.86E+01      960 1.94E-02      32.77    32.77  ********
ADVY2            1.53E+01      960 1.60E-02      26.95    59.72  ******
CALUVI           4.66E+00       11 4.24E-01       8.20    79.53  **
CHTRUV           2.60E+00       11 2.36E-01       4.57    84.10  *
CHTRTG           2.26E+00       11 2.06E-01       3.98    88.09
CONUKM           1.71E+00       11 1.56E-01       3.02    91.10
INITER           1.53E+00      480 3.19E-03       2.70    93.80
ADVEC            1.51E+00      480 3.15E-03       2.66    96.46
INICYCL          1.20E+00       10 1.20E-01       2.11    98.58
FINCYCL          2.35E-01       10 2.35E-02       0.41    98.99
SLIMCAT          1.56E-01        1 1.56E-01       0.28    99.27
FINITER          1.53E-01      480 3.19E-04       0.27    99.53
INIEXP           1.38E-01        1 1.38E-01       0.24    99.78
CHIMIE           7.49E-02      480 1.56E-04       0.13    99.91
CALFLUK          5.17E-02       11 4.70E-03       0.09   100.00
FINEXP           8.27E-05        1 8.27E-05       0.00   100.00
CALNUM           8.05E-05        1 8.05E-05       0.00   100.00
GENGRID          3.49E-05        1 3.49E-05       0.00   100.00
INICSTE          3.35E-06        1 3.35E-06       0.00   100.00
================================================================================
Totals           5.08E+01     6321
```

The 2 Prather advection routines (x and y directions) are the most expensive. The next 3 most expensive routines (`CALUVI, CHTRUV, CHTRTG`) simply interpolate the UKMO winds to the SLIMCAT grid. These could

probably be made more efficient.

17.3 SLIMCAT - Cray YMP8

Below is a flowtrace from a 5 day multilevel run with `NIV=3, LON=128, LAT=64` with (6-hourly) ECMWF winds and `LVERT=.TRUE.`.

```
+ flowview -Luc

                    Flowtrace Statistics Report
            Showing Routines Sorted by CPU Time (Descending)
                    (CPU Times are Shown in Seconds)

Routine Name     Tot Time  # Calls Avg Time Percentage  Accum%
---------------- -------- -------- -------- ----------  -------
ADVX2            2.73E+01      480 5.70E-02      22.92    22.92  *****
ADVY2            2.28E+01      480 4.76E-02      19.15    42.07  ****
CALFLU           1.08E+01       21 1.13E+00      19.00    61.07  ***
INCIRA           1.40E+01       21 6.68E-01      11.76    73.83  **
ADVZ2            1.32E+01      480 2.74E-02      11.03    84.69  **
HRTCO2           5.03E+00     1344 3.75E-03       4.22    88.91  *
SOLAR            2.46E+00     1344 1.83E-03       2.06    90.97
HRTO3            2.17E+00     1344 1.61E-03       1.82    92.79
INITER           2.07E+00      240 8.61E-03       1.73    94.52
FINCYCL          1.41E+00       20 7.04E-02       1.18    95.71
INTRAN           1.30E+00        1 1.30E+00       1.09    96.80
HTRH2O           8.10E-01     1344 6.03E-04       0.68    97.47
ADVEC            7.84E-01      240 3.27E-03       0.66    98.13
SETUSS           7.61E-01     1344 5.66E-04       0.64    98.77
INIEXP           2.85E-01        1 2.85E-01       0.24    99.01
FINITER          2.72E-01      240 1.13E-03       0.23    99.24
CHIMIE           2.38E-01      240 9.92E-04       0.20    99.44
INICYCL          2.17E-01       20 1.08E-02       0.18    99.62
THERML           1.11E-01     1344 8.27E-05       0.09    99.71
PCMDRD           9.76E-02       21 4.65E-03       0.08    99.79
CHTRON           7.54E-02     4071 1.85E-05       0.06    99.86
SOLANG           6.68E-02     1344 4.97E-05       0.06    99.91
SLIMCAT          6.03E-02        1 6.03E-02       0.05    99.96
PCMRAD           3.84E-02     1344 2.86E-05       0.03    99.99
CORPOLE          5.65E-03       21 2.69E-04       0.00   100.00
ORBIT            2.25E-04       21 1.07E-05       0.00   100.00
WCALEN           8.23E-05       21 3.92E-06       0.00   100.00
CALNUM           8.18E-05        1 8.18E-05       0.00   100.00
FINEXP           8.15E-05        1 8.15E-05       0.00   100.00
GENGRID          3.55E-05        1 3.55E-05       0.00   100.00
INICSTE          3.64E-06        1 3.64E-06       0.00   100.00
================================================================================
Totals           1.19E+02    17417
```

Again the Prather advection schemes account for about 53% of the total cost. The subroutine `INCIRA` is relatively expensive as it just interpolates the CIRA O_3 and T fields onto the SLIMCAT model levels. In general the MIDRAD scheme is well coded and cheap. Overall the above job ran at 72 Mflops.

Considering the number of calculations involved in the Prather second order moment advection scheme the subroutines ADVX2 etc. are coded efficiently. These are the most computationally intensive routines and should dominate the total cost of the model. As can be seen from the flowtraces above they do. The rest of the model is also sufficiently efficient. Of course, when a chemistry scheme is coupled to the model this will then dominate the total cost.

17.4 TOMCAT - Green O3000

Below is a profile from a 1 day TOMCAT 'full' tropospheric chemistry run on the CSAR O3000 Green (16 CPU). The run had NIV=31, LON=64, LAT=32 with (6-hourly) ECMWF winds.

```
+ prof -lines a.out.pcsampx.m
------------------------------------------------------------------------
SpeedShop profile listing generated Mon Jul 26 11:35:50 2004
   prof -lines a.out.pcsampx.m
                  a.out (n64): Target program
                      pcsampx: Experiment name
              pc,4,10000,0:cu: Marching orders
              R12000 / R12010: CPU / FPU
                          500: Number of CPUs
                          400: Clock frequency (MHz.)
  Experiment notes--
          From file a.out.pcsampx.m:
       Caliper point 0 at target begin, PID 10765979
                     /tmp/jtmp/tmpdir_126594/a.out
       Caliper point 1 at exit(0)
------------------------------------------------------------------------
Summary of statistical PC sampling data (pcsampx)--
                       49455: Total samples
                     494.550: Accumulated time (secs.)
                        10.0: Time per sample (msecs.)
                           4: Sample bin width (bytes)
------------------------------------------------------------------------
Function list, in descending order by time
------------------------------------------------------------------------
 [index]      secs    %    cum.%   samples  function (dso: file, line)

     [1]   112.710  22.8%  22.8%     11271  prls (a.out: prog.f, 44844)
     [2]    69.030  14.0%  36.7%      6903  harwel (a.out: prog.f, 26656)
     [3]    45.620   9.2%  46.0%      4562  __mp_barrier_nthreads (libmp.so: mp_barrier.c, 271)
     [4]    37.240   7.5%  53.5%      3724  ftoy (a.out: prog.f, 35354)
     [5]    27.780   5.6%  59.1%      2778  jac (a.out: prog.f, 42894)
     [6]    23.950   4.8%  64.0%      2395  impact (a.out: prog.f, 38164)
     [7]    23.650   4.8%  68.7%      2365  __exp (libm.so: exp.c, 103)
     [8]    13.700   2.8%  71.5%      1370  __mpdo_advx2_1 (a.out: prog.f, 9071)
     [9]    12.620   2.6%  74.1%      1262  diffun (a.out: prog.f, 34321)
    [10]    12.210   2.5%  76.5%      1221  __mpdo_consom_1 (a.out: prog.f, 12777)
    [11]    11.970   2.4%  79.0%      1197  __pow (libm.so: pow.c, 256)
    [12]    11.860   2.4%  81.4%      1186  __mpdo_advy2_1 (a.out: prog.f, 9683)
    [13]    10.350   2.1%  83.4%      1035  acaceto (a.out: prog.f, 28479)
    [14]     9.410   1.9%  85.4%       941  __mp_barrier_master (libmp.so: mp_barrier.c, 363)
    [15]     8.070   1.6%  87.0%       807  __mpdo_advz2_1 (a.out: prog.f, 10228)
```

```
    [16]     5.850   1.2%  88.2%       585  radabs (a.out: prog.f, 72194)
    [17]     5.840   1.2%  89.3%       584  __powdi (libftn.so: pow_di.c, 49)
    [18]     3.190   0.6%  90.0%       319  fyinit (a.out: prog.f, 36795)
    [19]     3.130   0.6%  90.6%       313  asad_step (a.out: prog.f, 32437)
    [20]     3.060   0.6%  91.2%       306  __read (libc.so.1: malloc.c, 907; compiled in read.s)
    [21]     2.900   0.6%  91.8%       290  memcpy (libc.so.1: stat.c, 32; compiled in bcopy.s)
    [22]     2.530   0.5%  92.3%       253  __mpdo_chimie_1 (a.out: prog.f, 18622)
    [23]     2.470   0.5%  92.8%       247  _defgu2sd (libffio.so: defgu2s.c, 243)
    [24]     2.370   0.5%  93.3%       237  lubksb2 (a.out: prog.f, 17728)
    [25]     2.250   0.5%  93.8%       225  __mpdo_calflu_17 (a.out: prog.f, 4876)
    [26]     1.980   0.4%  94.2%       198  __log (libm.so: log.c, 207)
    [27]     1.610   0.3%  94.5%       161  __write (libc.so.1: flush.c, 58; compiled in write.s)
    [28]     1.490   0.3%  94.8%       149  fyfixr (a.out: prog.f, 36286)
    [29]     1.330   0.3%  95.1%       133  fft991 (a.out: prog.f, 15642)
    [30]     1.190   0.2%  95.3%       119  _get_value (libfortran.so: lread.c, 735)
    [31]     1.050   0.2%  95.5%       105  calflu (a.out: prog.f, 3925)
    [32]     1.010   0.2%  95.7%       101  rpassm (a.out: prog.f, 16625)
    [33]     0.910   0.2%  95.9%        91  spetru1 (a.out: prog.f, 79519)
    [34]     0.870   0.2%  96.1%        87  _ld_read (libfortran.so: lread.c, 239)
    [35]     0.810   0.2%  96.3%        81  calcpv (a.out: prog.f, 15296)
    [36]     0.750   0.2%  96.4%        75  qpassm (a.out: prog.f, 15875)
    [37]     0.730   0.1%  96.6%        73  presetr (a.out: prog.f, 78043)
    [38]     0.720   0.1%  96.7%        72  pharwel (a.out: prog.f, 26080)
    [39]     0.680   0.1%  96.8%        68  __mpdo_wdepin_4 (a.out: prog.f, 21008)
    [40]     0.660   0.1%  97.0%        66  __mp_wait_for_completion (libmp.so: mp.c, 883)
    [41]     0.660   0.1%  97.1%        66  ludcmp2 (a.out: prog.f, 17760)
    [42]     0.620   0.1%  97.2%        62  calsub (a.out: prog.f, 12858)
    [43]     0.600   0.1%  97.3%        60  quanto13 (a.out: prog.f, 29156)
    [44]     0.460   0.1%  97.4%        46  vdiff (a.out: prog.f, 84934)
    [45]     0.440   0.1%  97.5%        44  __mpdo_physics_2 (a.out: prog.f, 70943)
    [46]     0.420   0.1%  97.6%        42  minv (a.out: prog.f, 17673)

<snip>

           494.550 100.0% 100.0%     49455  TOTAL
```